

MARCH 1988

COMPUTER





GET ON THE "PAWS™ EXPRESSWAY"



TO SYSTEM DESIGN SOLUTIONS

Why not take the shortest and fastest route toward system design solutions? To define and model a large, complex system, you need a modeling tool that can address the system's total architecture.



PAWS/GPSM™ is an easy-to-use high productivity modeling and simulation tool that lets you evaluate alternative architectures while focusing on performance. It provides a state-of-the-art environment for developing accurate and reliable product definitions.



PAWS provides high level primitives closely resembling the user's intuition. PAWS models the behavior of a total system implemented in diverse technologies including software, hardware, and firmware.



GPSM, the graphical interface to PAWS, helps you quickly and easily transfer your ideas into pictures. GPSM helps you visualize multiple data and control flows through several components and incrementally synthesize models of a total system.



PAWS/GPSM is flexible. It lets you refine your model as your system design evolves so you can eliminate potential problems as early as possible in the product design cycle.

Call (512) 474-4526 to receive information on PAWS/GPSM. And get on the PAWS expressway to system design solutions.

PAWS and GPSM are trademarks of Information Research Associates, Inc.

IRA

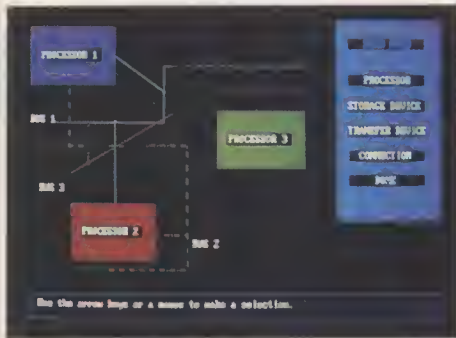
Information Research Associates • 911 W. 29th St. • Austin, Texas 78705 • (512) 474-4526

Reader Service Number 1

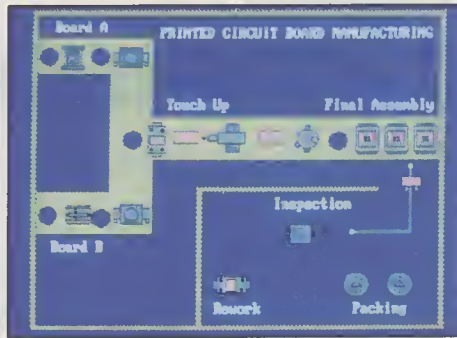
Breakthrough in presentation of simulation results

SIMSCRIPT II.5 with SIMANIMATION

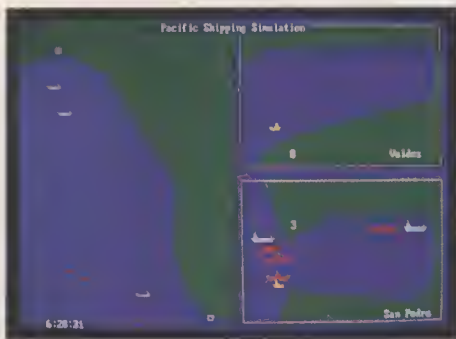
Now you see an animated picture of the system



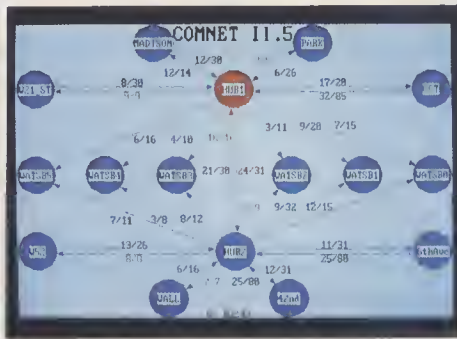
Computer communications-NETWORK II.5



Factory in operation-SIMFACTORY



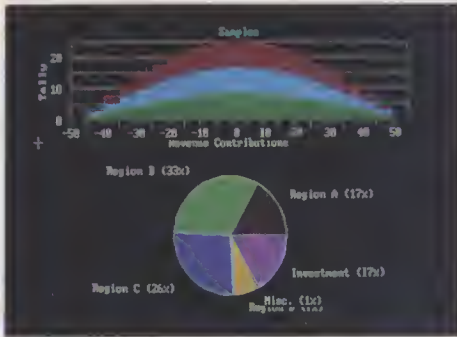
Transportation network



Telecommunications-COMNET II.5



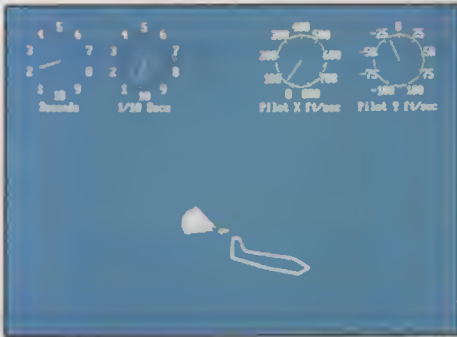
Military operation



Presentation graphics



Mining operation in process



Pilot ejection

Free trial and training

See for yourself how simulation results are now easier to understand.

The free trial contains everything you need to try SIMANIMATION® on your computer.

We send you SIMSCRIPT II.5, animated models, and complete documentation. You can build your own model or modify one of ours.

Try the SIMSCRIPT II.5 language, the timeliness of our support, the accuracy of our documentation, and the facilities for error checking--**everything you need for a successful project.**

No cost, no obligation.

Act now for free training

For a limited time we will also include free training.

For immediate information

Call Hal Duncan at (619) 457-9681. In the UK call Richard Eve on (01) 940-3606.

Rush information on SIMSCRIPT II.5 with SIMANIMATION

Free trial-learn the reasons for the broad and growing popularity of SIMSCRIPT II.5 with SIMANIMATION.

No cost, no obligation.

Limited offer-return the coupon today and we will also include one free course enrollment worth \$950.

Name

Organization

Address

City

State

Zip

Telephone

Computer

Operating System

IEEE COMP

Return to:

CACI
3344 North Torrey Pines Court
La Jolla, California 92037
**Or, better yet, call Hal Duncan at
(619) 457-9681**

In the UK

CACI
26 The Quadrant
Richmond, Surrey, England TW9 1DL
Call Richard Eve on (01) 940-3606

COMPUTER

March 1988

Published by the Computer Society

Vol. 21, No. 3

ARTICLES

8 Guest Editor's Introduction: Artificial Neural Systems

Bruce D. Shriver

11 The "Neural" Phonetic Typewriter

Teuvo Kohonen

Based on a neural network processor for the recognition of phonetic units of speech, this speaker-adaptive system transcribes dictation using an unlimited vocabulary.

25 Neural Nets for Adaptive Filtering and Adaptive Pattern Recognition

Bernard Widrow and Rodney Winter

A new multilayer adaptation algorithm that descrambles output and reproduces original patterns is advancing the practicality of neural-network pattern-recognition systems.

41 VLSI Implementation of a Neural Network Model

Hans P. Graf, Lawrence D. Jackel, and Wayne E. Hubbard

To obtain the full benefit of neural network algorithms, we need special-purpose hardware. An experimental CMOS VLSI circuit was tested as an associative memory and as a pattern classifier.

52 Computing Motion Using Analog and Binary Resistive Networks

James Hutchinson, Christof Koch, Jin Luo, and Carver Mead

We can compute optical flow by injecting currents into resistive networks and recording the stationary voltage distribution at each node.

65 A Neural Network for Visual Pattern Recognition

Kunihiko Fukushima

This model of the neural network offers insight into the brain's complex mechanisms as well as design principles for new information processors.

77 The ART of Adaptive Pattern Recognition by a Self-Organizing Neural Network

Gail A. Carpenter and Stephen Grossberg

The adaptive resonance theory suggests a solution to the stability-plasticity dilemma facing designers of learning systems.

91 Computing with Structured Neural Networks

Jerome A. Feldman, Mark A. Fandy, and Nigel H. Goddard

Designing and debugging massively parallel, connectionist systems will require appropriate tools. One such package is now in use at Rochester and other laboratories.

105 Self-Organization in a Perceptual Network

Ralph Linsker

How can a perceptual system develop to recognize specific features of its environment, without being told which features it should analyze, or even whether its identifications are correct?

Circulation: *Computer* (ISSN 0018-9162) is published monthly by the Computer Society of the IEEE. IEEE Headquarters: 345 East 47th St., New York, NY 10017; Computer Society of the IEEE, Publications Office, 10662 Los Vaqueros Circle, Los Alamitos, CA 90720; (714) 821-8380. Annual subscription included in society member dues. Nonmember prices: available upon request. Single copy prices: members \$10.00; nonmembers \$20.00. This magazine is also available in microfiche form.

Postmaster: Send undelivered copies and address changes to *Computer*, IEEE Service Center, 445 Hoes Lane, Piscataway, NJ 08854. Second class postage is paid at New York, New York, and at additional mailing offices.

Copyright and reprint permissions: Copyright © 1988 by the Institute of Electrical and Electronics Engineers, Inc. All rights reserved. Abstracting is permitted with credit to the source. Libraries are permitted to photocopy beyond the limits of US copyright law for private use of patrons: (1) those post-1977 articles that carry a code at the bottom of the first page, provided the per-copy fee indicated in the code is paid through the Copyright Clearance Center, 29 Congress Street, Salem, MA 01970. (2) pre-1978 articles without fee. Instructors are permitted to photocopy isolated articles for noncommercial classroom use without fee. For other copying, reprint, or republication permission, write to True Seaborn, Editor and Publisher, *Computer*, 10662 Los Vaqueros Circle, Los Alamitos, CA 90720. All rights reserved. Copyright © 1988 by the Institute of Electrical and Electronics Engineers, Inc.

Editorial: Unless otherwise stated, bylined articles, as well as product and service descriptions, reflect the author's or firm's opinion. Inclusion in *Computer* does not necessarily constitute endorsement by the IEEE or the Computer Society. All submissions are subject to editing for style, clarity, and space considerations. *Computer* subscribes to the Computer Press Association's code of professional ethics.



DEPARTMENTS

6 President's Message

Our goals for 1988

119 Standards

122 Computer Society News

1988 TC chairs

124 Update

128 New Product Reviews

Alternative input devices; entry-level network; Living C

134 New Products

138 IC Announcements

140 Microsystem Announcements

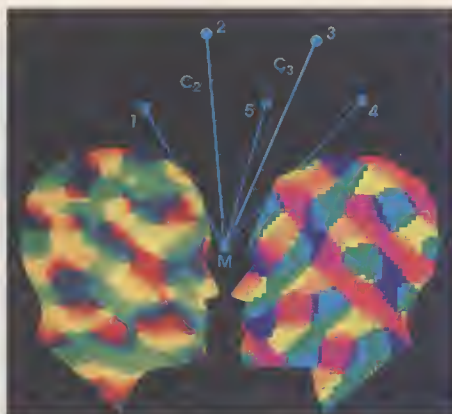
143 Conferences

146 Calendar

148 Call for Papers

155 Book Reviews

159 New Literature



The visual cortex contains many cells that respond selectively to the orientation of a light-dark edge. The cover shows (left) a color coding of the orientation-selective cells in monkey visual cortex, measured by Blasdel and Salama using a voltage-sensitive dye method, and (right) a computer-generated pattern of such cells that develops in Linsker's mathematical network (see his article in this issue, p. 105). The patterns show qualitative similarities.

Cover images courtesy of Gary Blasdel and Guy Salama (first published in *Nature*, June 5, 1986, p. 583, Figure 6a) and Ralph Linsker

Cover design: Jay Simpson, Design & Direction

In the next issue

Multiple-valued logic

Computer Society Information Page, 142; Membership Application Form, 133; Career Opportunities, 149; Advertising and Product Indexes, 160; Reader Service Card, 160A

COMPUTER

10662 Los Vaqueros Circle, Los Alamitos, CA 90720

Editor-in-Chief

Bruce D. Shriver, IBM T.J. Watson Research Center

Editorial Board

Dharma P. Agrawal, North Carolina State University
 Scott Davidson, AT&T
 Richard Eckhouse, MOCO
 Gideon Frieder, Syracuse University
 Edmund Gallizzi, Eckerd College
 John Gurd, University of Manchester
 James H. Haynes, University of California, Santa Cruz
 Wiley McKinzie, Rochester Institute of Technology
 K. M. Mohiuddin, IBM Almaden Research Laboratory
 William Riddle, Software Productivity Consortium
 Sallie Sheppard, Texas A&M University
 Helen M. Wood, National Bureau of Standards
 Akinori Yonezawa, Tokyo Institute of Technology

Submissions

Send six double-spaced copies of articles and special-issue proposals to Bruce D. Shriver, IBM T.J. Watson Research Center, PO Box 704, Yorktown Heights, NY 10598; (914) 789-7626. For electronic submission, Shriver's addresses are Comppail + , b.shriver; CSnet, Shriver@ibm.com; Vnet, shriver at yktvmh.

Staff

Editor and Publisher: True Seaborn
Assistant Publisher: Douglas L. Combs
Assistant to the Publisher: Pat Paulsen
Managing Editor: Marilyn Potes
Issue Editor: Nancy Hays
Assistant Editor: Chuck Governale, Steve H. Wilcox
Art Director: Jay Simpson
Design/Production: Miriam Wiegel
Membership Manager: Christina Champion
Circulation Manager: Paul Zive
Advertising Director: Dawn Peck
Advertising Coordinators: Heidi Rex, Marian Tibayan

Magazine Advisory Committee

Sallie Sheppard (chair), Sumit DasGupta, Michael Evangelist, James J. Farrell III, Ted Lewis, David Pessel, True Seaborn, Bruce D. Shriver, John Staudhammer

Publications Board

James H. Aylor (chair), Victor Basili, J. Richard Burke, Jon T. Butler, J. T. Cain, Sumit DasGupta, Michael Evangelist, James J. Farrell III, Glen G. Langdon, Ted Lewis, Ming T. Liu, Theo Pavlidis, David Pessel, Sallie Sheppard, Bruce D. Shriver, Charles B. Silio, John Staudhammer, Harold Stone, Steven L. Tanimoto

Taking The Lead In Applied R&D

C O N T E L • T E C H N

***"If there's a better way,
we'll find it."***



Dr. Alan Salisbury, Director
Contel Technology Center

A Diversified Company, With One Vision.

At Contel Corporation, finding the better way is our goal. It's also our business. And our efforts have resulted in over 25 years of leadership in the telecommunications and information services industry. As one of the youngest companies ever admitted to the New York Stock Exchange, we've grown through hundreds of acquisitions to become a \$3-billion, Forbes 500 corporation. We also rank as the third-largest independent telephone company in the U.S. It's exciting to be out in front.

And Ready To Take The Lead.

That's why we've taken the initiative with a brand new Technology Center responsible for the transfer of Applied R&D technology to products and services used by all of our operating companies and customers. We'll be turning emerging technologies into useful solutions. And bridging the gap between research and application.

With Imaginations At Work.

That's what you'll find here. The best minds in the business. Leading Scientists and Engineers creating innovative and practical solutions for better telecommunications and information systems. Those who appreciate the freedom to design their own projects. And influence our future

direction. If you're looking to turn *your* ideas into realities, you can do it here. We're building four initial laboratories, yet, we offer the opportunity and challenge of crossing technology lines.

Artificial Intelligence/ Man-Machine Interface Laboratory



This laboratory is involved in the application of AI and Expert System techniques to the broad class of

telecommunications and information systems problems, and pursues improved man-machine interfaces for these systems as well. Typical project areas include speech recognition, large-screen displays and optimal workstation configurations.



© Copyright 1988 Contel Corporation.

Our goals for 1988

In this month's column, I want to report the results of the Executive Committee's January 6-7 planning meeting. As stated in my January message, I intend to capitalize on the various self-assessment studies completed by our boards in 1987 and to expand the Computer Society's role in international activities. In addition, the society will take the necessary steps to open its Asian Office in Tokyo during the first part of this year.

To this end, I gave the following charge to Executive Committee members prior to the planning meeting:

Present an overview of accomplishments desired for 1988. These should result in (1) implementing the principal recommendations from the self-assessment reports, (2) enhancing international activities and relations, (3) raising the visibility of the Computer Society, (4) increasing the society's presence at major conferences, (5) raising staff effectiveness, (6) expanding the use of appropriate technology throughout the society (electronic mail, electronic media, etc.), and (7) increasing the emphasis on the awards program.

The Executive Committee spent nearly two days responding to this charge. The discussion following each presentation attempted to define the requisite actions to be taken by each officer. At the end of the meeting, the group finalized a list of objectives for each board. The associated vice presidents will carry this information to their boards for further consideration at meetings held during Compcon Spring 88 in San Francisco. A full activity plan, complete with performance measures and appropriate milestones, will be developed by the boards and be available for presentation at the Executive Committee meeting on March 3. Some of the major issues may also be placed on the agenda for the Board of Governor's caucus the previous evening.

The list of objectives is given below.



Edward A. Parrish, Jr.

Global goal:

- Make the Computer Society the No. 1 organization for computing professionals worldwide.

Education Activities Board:

- Work to develop new programs and initiatives.
- Interact with the Area Activities Board to help with development of student chapters.

Area Activities Board:

- Enhance programs directed at students.
- Serve computer science as well as electrical engineering and computer engineering students.
- Develop a comprehensive tutorial program.

Membership and Information Board:

- Continue promotions and other activities to reach membership goal of 94,500.
- Develop generalized promotion techniques.
- Address non-US membership pool.
- Strengthen and expand information activities.

Publications Board:

- Expand and enhance CS Press activities.
- Control/contain costs.

- Exploit electronic media—for example, bulletin boards and news groups.

Standards Activities Board:

- Consolidate activities.
- Expand international activities.
- Refine international procedures.

Conferences and Tutorials Board:

- Enhance quality of products.
- Consider experiment with individualized, track-oriented conference proceedings.
- Consider board reorganization.

Technical Activities Board:

- Develop quality products (specialized publications, electronic news groups, etc.) for technical committee members.
- Consider implementation of membership application program.

Issues for all boards:

- Quality of programs and activities.
- Membership promotion and retention.
- Attention to students and new graduates.
- Promotion and enhancement of the Computer Society's image among technical professionals.
- Financial impact and planning.
- Membership communications.

Considerations for the Awards Committee:

- New, high prestige awards.
- Awards for student work—select best three to five dissertations in computer engineering and computer science and publish through CS Press.
- Outstanding Young Computer Scientist/Engineer Award.

At the conclusion of the meetings at Compcon Spring 88, each board will have established its activity plan for the year, along with appropriate milestones and success measures. This should help provide continuity of purpose from meeting to meeting and involve each entity in establishing the society's agenda.

Watch for progress reports in this column in the months ahead.

Edward A. Parrish, Jr.
Computer Society President

IEEE INTERNATIONAL CONFERENCE ON COMMUNICATIONS '88

DIGITAL TECHNOLOGY...

SPANNING THE UNIVERSE

June 12-15, 1988 at The Wyndham Franklin Plaza Hotel, Philadelphia, Pennsylvania

ICC '88...an international conference bringing together the world's knowledge in communications management and technology in the birthplace of America.

A comprehensive technical program will be offered complimented by an exciting social program. The **technical program** will feature tours of local industry; a symposium on Open Network Architecture (ONA); an evening "Quality" session featuring RBOC and industry executives; an exhibition hall providing attendees with an opportunity to mix with exhibitors' representatives; and, of course, a broad "tracked" series of technical sessions spotlighting: • **Optical technologies** • **Intelligent networks** • **Data communication techniques** • **Radio systems** • **Operations, performance and quality.**

ICC '88 will also offer keynote speakers at a conference banquet and an IEEE awards luncheon, as well as a special **Social program** highlighted by a Philadelphia-Style Block Party; Tours of Historic Philadelphia and the Land of the Pennsylvania Dutch; a Dinner Cruise; and a great casino excursion to Atlantic City.



Sponsors: IEEE Communications Society and the Philadelphia Section IEEE

For More Information Contact ICC '88 by phone...

1-800-ICC88PH (in the continental U.S.) (215) 972-1308 (outside U.S.)

weekdays between 8 A.M. and 4:30 P.M. (E.S.T.) or complete the coupon below and mail...

CLIP AND MAIL TO

ICC '88, c/o ATT Network Systems,
1800 John F. Kennedy Blvd., Suite 1300,
Philadelphia, PA 19103

- ☐ Please send more information and registration for ICC '88
☐ Please send information for ICC '88 exhibitors

NAME: _____

COMPANY: _____

ADDRESS: _____

PHONE: _____

Guest Editor's Introduction

Artificial Neural Systems



Bruce D. Shriver

IBM T.J. Watson Research Center

Conventional digital computers are extremely good at executing sequences of instructions that have been precisely formulated for them, with the "stored program" representing the processing steps that need to be done. The human brain, on the other hand, performs well at such tasks as vision, speech, information retrieval, and complex spatial and temporal pattern recognition in the presence of noisy and distorted data—tasks that are very difficult for sequential digital computers to do at all. How does the brain accomplish this, given that its

"processing elements" (neurons) are significantly slower than the processing elements of contemporary supercomputers? Neurons, which are electrochemical devices, can respond in milliseconds, whereas current, off-the-shelf electronic

technology can switch states in nanoseconds.

Current estimates place the number of neurons in the human brain at 10^{11} . They are organized in a complex, unknown interconnection structure, and individual neurons may be connected to several thousand other neurons. It is not yet understood how this massively parallel interconnected system of neurons (a "biological neural network") allows us to store, represent, retrieve, and manipulate data such as images, smells, sensations, and thoughts. We do not know how it

represents a person's face, for example, so that merely seeing someone's eyes allows us to formulate a complete image in which we recall other important personal information—such as the way he or she walks. We do not know how we store equations, how we manipulate ideas without writing them down, or how we learn to speak, see, and hear. Yet, we do all of these things amazingly well.

Once vision and speech, for example, are well-enough understood to be reduced to algorithmic form, they can be realized on conventional digital computers just as well as on artificial neural systems. At that point, it will be a cost/performance issue as to which technology or combination of technologies will be employed to realize these capabilities.

Both the literature and the number of professional society meetings focusing on artificial neural systems are growing at an amazing rate. A number of technical disciplines are involved in the wide variety of independent industrial, government, and university-based activities and studies. Neurobiologists, neurophysiologists, mathematicians, physicists, psychologists, computer scientists, and engineers are studying and formulating theories about how computations actually occur in nature.

Some researchers are undertaking these studies so as to ultimately understand how the brain works and thus closely adhere to or attempt to understand the biology involved. Others are evolving entirely new computation paradigms based on the simple models that are part of the new theories. Some of the new paradigms are best termed "biologically influenced" because they involve assumptions that are not biologically accurate. These biologically influenced computational paradigms have been used to solve difficult optimization problems and to implement associative memories.¹ Although the field of artificial neural systems has roots going back over 25 years,^{1,2} there currently is no consensus of what is important to study or how to go about studying it. Some researchers are combining conventional AI's symbolic and heuristic approach to complex problem solving with the subsymbolic approach, where neural models apparently perform well.³

This special issue of *Computer* is entitled "Artificial Neural Systems"—that is, "artificial" as opposed to the "biological" neural systems appearing in nature. The study of artificial neural systems goes under the guise of many names in the lit-

erature: neural networks, connectionist models, parallel distributed processing models, layered self-adaptive systems, and self-organizing systems. Terms like neurocomputers, neuromorphic systems, netware, and cyberware are being introduced into our technical "jargon." For a good introduction to computing with artificial neural systems—one that reviews six important neural models used for pattern classification—see Lippmann.²

This issue is exclusively dedicated to a selection of interesting and important work in several areas of artificial neural systems. Because workers in the field are exploring many different areas, the articles reflect the diversity and robustness of these interests. Some of the authors closely adhere to the biology involved, while others are developing biologically influenced models and systems. Results are presented that are based on

(1) using the most elementary model of a neuron (one that sums its N "weighted" inputs and outputs the result through a nonlinearity),

(2) interconnecting these "simple neurons" in a network topology involving feedback, and

(3) employing various rules by which weights are adjusted (that is, the way in which learning—self-adaptation, self-organization—occurs).

Applications such as visual pattern recognition, speech recognition, motion detection, adaptive pattern recognition, as well as VLSI and simulation implementations of artificial neural systems are presented.

I hope you enjoy reading these articles. Clearly, this newly reincarnated field has some interesting and promising results to share, but it is not known how these results will scale up to more real-world related tasks. Numerous important research questions emerge from even this small sampling of articles. For example: Are specific models more appropriate for given classes of computations than other models? How does the sample set of learning situations affect the resulting characteristics of the neural system both during training and during operation? How can supervised and unsupervised learning be combined in such systems? How do the various interconnection structures affect the computational and operational characteristics of the system? What hardware is best for supporting the particular neural-network models? What algorithms can be formulated using the massively parallel paradigm of neural networks? And on and on...□

Acknowledgment

I am indebted to both the authors and the referees for the extraordinary effort put forth in developing this issue.

References

1. David W. Tank and John J. Hopfield, "Collective Computation in Neuronlike Circuits," *Scientific American*, Dec. 1987, pp. 104-114.
2. Richard P. Lippman, "An Introduction to Computing with Neural Nets," *IEEE ASSP Magazine*, Apr. 1987, pp. 4-22.
3. Scott E. Fahlman and Geoffrey E. Hinton, "Connectionist Architectures for Artificial Intelligence," *Computer*, Jan. 1987, pp. 100-109.



Bruce D. Shriver, a research staff member at IBM's T.J. Watson Research Center in Yorktown Heights, New York, is currently editor-in-chief of *Computer*. Before that, he was the first editor-in-chief of *IEEE Software* and, for five years, had served as an associate technical editor of *Computer*. For 11 years, before joining IBM in June 1984, Shriver was the Alfred Lamson Research Professor of Computer Science at the University of Southwestern Louisiana. He has published over 50 technical papers and chaired some 14 dissertation committees.

Shriver is a senior member of IEEE and a member of the Computer Society's Board of Governors, Publications Board, and Magazine Advisory Committee. He is a past chair of the Technical Committee on Microprogramming and a past vice chair of the TC on Computational Medicine.

He received his PhD in computer science from the State University of New York at Buffalo in 1971.

Shriver's address is IBM T.J. Watson Research Center, PO Box 704, Yorktown Heights, NY 10598.

IEEE INFOCOM '89

The Conference on Computer Communications

Technology: Emerging or Converging?

April 24-27, 1989 • Westin Hotel, Ottawa, Ontario, Canada

CALL FOR PAPERS

**Eighth Annual Joint Conference of the
IEEE Computer and Communications Societies**

Sponsored by the Technical Committees for Computer Communications of the Societies

Authors are invited to submit full papers on recent advances in computer and communications disciplines. Areas of interest include, but are not limited to:

- Telecommunications and Switching
- Integrated Services Networks
- Multiple Access Algorithms
- Satellite Communications Networks
- Network Architectures
- Local Area Networks
- Metropolitan and Wide Area Networks
- Network Management
- Network Modeling and Analysis
- Performance Evaluation
- Packet Radio
- ISDN Technology
- Communication Protocols
- Network Addressing and Routing
- Database Systems
- Concurrency Control
- Network Interconnection
- Computer Security and Privacy
- Network Reliability
- Local Distribution Loops
- Office Automation
- Network Standards

SCHEDULE

Full Paper (5 copies) — August 1, 1988
Notification of Acceptance — October 28, 1988
Camera Ready Copy — December 19, 1988
Conference — April 25-27, 1989
Tutorials — April 24, 1989

*Submit papers to: Prof. J.W. Mark, Technical Program Chairman, IEEE INFOCOM '89
Dept. of Electrical Engineering, University of Waterloo, Waterloo, Ontario, Canada N2L 3G1
Telephone (519) 888-4016*



THE COMPUTER SOCIETY
OF THE IEEE



THE INSTITUTE OF ELECTRICAL
AND ELECTRONICS ENGINEERS, INC.



IEEE COMMUNICATIONS SOCIETY

The "Neural" Phonetic Typewriter

Teuvo Kohonen

Helsinki University of Technology

In 1930 a Hungarian scientist, Tihamér Nemes, filed a patent application in Germany for the principle of making an optoelectrical system automatically transcribe speech. His idea was to use the optical sound track on a movie film as a grating to produce diffraction patterns (corresponding to speech spectra), which then could be identified and typed out. The application was turned down as "unrealistic." Since then the problem of automatic speech recognition has occupied the minds of scientists and engineers, both amateur and professional.

Research on speech recognition principles has been pursued in many laboratories around the world, academic as well as industrial, with various objectives in mind.¹ One ambitious goal is to implement automated query systems that could be accessed through public telephone lines, because some telephone companies have observed that telephone operators spend most of their time answering queries. An even more ambitious plan, adopted in 1986 by the Japanese national ATR (Advanced Telecommunication Research) project, is to receive speech in one language and to synthesize it in another, on line. The dream of a phonetic typewriter that can produce text from arbitrary dictation is an old one; it was envisioned by Nemes and is still being pursued today. Several dozen devices, even special microcircuits, that can recognize isolated

Based on a neural network processor for the recognition of phonetic units of speech, this speaker-adaptive system transcribes dictation using an unlimited vocabulary.

words from limited vocabularies with varying accuracy are now on the market. These devices have important applications, such as the operation of machines by voice, various dispatching services that employ voice-activated devices, and aids for seriously handicapped people. But in spite of big investments and the work of experts, the original goals have not been reached. High-level speech recognition has existed so far only in science fiction.

Recently, researchers have placed great hopes on artificial neural networks to perform such "natural" tasks as speech recognition. This was indeed one motivation for us to start research in this area many years ago at Helsinki University of Technology. This article describes the result of that research—a complete "neural" speech recognition system, which recognizes phonetic units, called *phonemes*, from a continuous speech signal. Although motivated by neural network principles, the choices in its design must be regarded as a compromise of many technical aspects of those principles. As our system is a genuine "phonetic typewriter" intended to transcribe orthographically edited text from an unlimited vocabulary, it cannot be directly compared with any more conventional, word-based system that applies classical concepts such as dynamic time warping¹ and hidden Markov models.²

Why is speech recognition difficult?

Automatic recognition of speech belongs to the broader category of pattern recognition tasks,³ for which, during the past 30 years or so, many heuristic and even sophisticated methods have been tried. It may seem strange that while progress in many other fields of technology has

been astoundingly rapid, research investments in these "natural" tasks have not yet yielded adequate dividends. After initial optimism, the researchers in this area have gradually become aware of the many difficulties to be surmounted.

Human beings' recognition of speech consists of many tasks, ranging from the detection of phonemes from speech waveforms to the high-level understanding of messages. We do not actually hear all speech elements; we realize this easily when we try to decipher foreign or uncommon utterances. Instead, we continuously relate fragmentary sensory stimuli to contexts familiar from various experiences, and we unconsciously test and reiterate our perceptions at different levels of abstraction. In other words, what we believe we *hear*, we in fact *reconstruct* in our minds from pieces of received information.

Even in clear speech from the same speaker, distributions of the spectral samples of different phonemes overlap. Their statistical density functions are not Gaussian, so they cannot be approximated analytically. The same phonemes spoken by different persons can be confused too; for example, the /ε/ of one speaker might sound like the /n/ of another. For this reason, absolutely speaker-independent detection of phonemes is possible only with relatively low accuracy.

Some phonemes are spectrally clearer and stabler than others. For speech recognition purposes, we distinguish three acoustically different categories:

(1) Vocal (voiced, nonturbulent) phonemes, including the vowels, semivowels (/j/, /v/), nasals (/m/, /n/, /ŋ/), and liquids (/l/, /r/)

(2) Fricatives (/s/, /š/, /z/, etc.)

(3) Plosives (/k/, /p/, /t/, /b/, /d/, /g/, etc.)

The phonemes of the first two categories have rather well-defined, stationary spectra, whereas the plosives are identifiable only on the basis of their transient properties. For instance, for /k,p,t/ there is a silence followed by a short, faint burst of voice characteristic of each plosive, depending on its point of articulation (lips, tongue, palate). The transition of the speech signal to the next phoneme also varies among the plosives.

A high-level automatic speech recognition system also should interpret the semantic content of utterances so that it can maintain selective attention to particular portions of speech. This ability would call for higher thinking processes, not only

Machine interpretation of complete sentences has been accomplished only with artificially limited syntax.

imitation of the operation of the preattentive sensory system. The first large experimental speech-understanding systems followed this line of thought (see the report of the ARPA project,⁴ which was completed around 1976), but for commercial application such solutions were too expensive. Machine interpretation of the meaning of complete sentences is a very difficult task; it has been accomplished only when the syntax has been artificially limited. Such "party tricks" may have led the public to believe that practical speech recognition has reached a more advanced level than it has. Despite decades of intensive research, no machine has yet been able to recognize general, continuous speech produced by an arbitrary speaker, when no speech samples have been supplied.

Recognition of the speech of arbitrary speakers is much more difficult than generally believed. Existing commercial speaker-independent systems are restricted to isolated words from vocabularies not exceeding 40 words. Reddy and Zue estimated in 1983 that for speaker-independent recognition of connected speech, based on a 20,000-word vocabulary, a computing power of 100,000 MIPS, corresponding to 100 supercomputers, would be necessary.⁵ Moreover, the detailed programs to perform these operations have not been devised. The difficulties would be even greater if the vocabularies were unlimited, if the utterances were loaded with emotions, or if speech were produced under noisy or stressful conditions.

We must, of course, be aware of these difficulties. On the other hand, we would never complete any practical speech recognizer if we had to attack all the problems simultaneously. Engineering solutions are

therefore often restricted to particular tasks. For instance, we might wish to recognize isolated commands from a limited vocabulary, or to type text from dictation automatically. Many satisfactory techniques for speaker-specific, isolated-word recognition have already been developed. Systems that type English text from clear dictation with short pauses between the words have been demonstrated.⁶ Typing unlimited dictation in English is another intriguing objective. Systems designed for English recognize words as complete units, and various grammatical forms such as plural, possessive, and so forth can be stored in the vocabulary as separate word tokens. This is not possible in many other languages—Finnish and Japanese, for example—in which the grammar is implemented by inflections and there may be dozens of different forms of the same root word. For inflectional languages the system must construct the text from recognized phonetic units, taking into account the transformations of these units due to coarticulation effects (i.e., a phoneme's acoustic spectrum varies in the context of different phonemes).

Especially in image analysis, but in speech recognition too, many newer methods concentrate on structural and syntactic relationships between the pattern elements, and special grammars for their analysis have been developed. It seems, however, that the first step, preanalysis and detection of primary features such as acoustic spectra, is still often based on rather coarse principles, without careful consideration of the very special statistical properties of the natural signals and their clustering. Therefore, when new, highly parallel and adaptive methods such as artificial neural networks are introduced, we assume that their capacities can best be utilized if the networks are made to adapt to the real data, finding relevant features in the signals. This was in fact one of the central assumptions in our research.

To recapitulate, speech is a very difficult stochastic process, and its elements are not unique at all. The distributions of the different phonemic classes overlap seriously, and to minimize misclassification errors, careful statistical as well as structural analyses are needed.

The promise of neural computers

Because the brain has already implemented the speech recognition function

(and many others), some researchers have reached the straightforward conclusion that artificial neural networks should be able to do the same, regarding these networks as a panacea for such "natural" problems. Many of these people believe that the only bottleneck is computing power, and some even expect that all the remaining problems will be solved when, say, optical neural computers, with a vast computing capacity, become feasible. What these people fail to realize is that *we may not yet have discovered what biological neurons and neural systems are like*. Maybe the machines we call neural networks and neural computers are too simple. Before we can utilize such computing capacities, we must know *what* and *how* to compute.

It is true that intriguing simulations of new information-processing functions, based on artificial neural networks, have been made, but most of these demonstrations have been performed with artificial data that are separable into disjoint classes. Difficulties multiply when natural, stochastic data are applied. In my own experience the quality of a neural network must be tested in an on-line connection with a natural environment. One of the most difficult problems is dealing with input data whose statistical density functions overlap, have awkward forms in high-dimensional signal spaces, and are not even stationary. Furthermore, in practical applications the number of samples of input data used for training cannot be large; for instance, we cannot expect that every user has the patience to dictate a sufficient number of speech samples to guarantee ultimate accuracy.

On the other hand, since digital computing principles are already in existence, they should be used wherever they are superior to biological circuits, as in the syntactic analysis of symbolic expressions and even in the spectral analysis of speech waveforms. The discrete Fourier transform has very effective digital implementations.

Our choice was to try neural networks in a task in which the most demanding statistical analyses are performed—namely, in the optimal detection of the phonemes. In this task we could test some new learning methods that had been shown to yield a recognition accuracy comparable to the decision-theoretic maximum, while at the same time performing the computations by simple elements, using a minimal amount of sample data for training.

In practical neural-network applications, the number of input samples used for training cannot be large.

Acoustic preprocessing

Physiological research on hearing has revealed many details that may or may not be significant to artificial speech recognition. The main operation carried out in human hearing is a frequency analysis based on the resonances of the basilar membrane of the inner ear. The spectral decomposition of the speech signal is transmitted to the brain through the auditory nerves. Especially at lower frequencies, however, each peak of the pressure wave gives rise to separate bursts of neural impulses; thus, some kind of time-domain information also is transmitted by the ear. On the other hand, a certain degree of synchronization of neural impulses to the acoustic signals seems to occur at all frequencies, thus conveying phase information. One therefore might stipulate that the artificial ear contain detectors that mimic the operation of the sensory receptors as fully as possible.

Biological neural networks are able to enhance signal transients in a nonlinear fashion. This property has been simulated in physical models that describe the mechanical properties of the inner ear and chemical transmission in its neural cells.^{7,8} Nonetheless, we decided to apply conventional frequency analysis techniques, as such, to the preprocessing of speech. The main motivations for this approach were that the digital Fourier analysis is both accurate and fast and the fundamentals of digital filtering are well understood. Standard digital signal processing has been considered sufficient in acoustic engineering and telecommunication. Our decision was thus a typical engineering choice. We also believed the self-organizing neural

network described here would accept many alternative kinds of preprocessing and compensate for modest imperfections, as long as they occur consistently. Our final results confirmed this belief; at least there were no large differences in recognition accuracies between stationary and transient phonemes.

Briefly, the complete acoustic preprocessor of our system consists of the following stages:

- (1) Noise-canceling microphone
- (2) Preamplifier with a switched-capacitor, 5.3-kHz low-pass filter
- (3) 12-bit analog-to-digital converter with 13.02-kHz sampling rate
- (4) 256-point fast Fourier transform, computed every 9.83 ms using a 256-point Hamming window
- (5) Logarithmization and filtering of spectral powers by fourth-order elliptic low-pass filters
- (6) Grouping of spectral channels into a 15-component real-pattern vector
- (7) Subtraction of the average from all components
- (8) Normalization of the resulting vector into constant length

Operations 3 through 8 are computed by the signal processor chip TMS 32010 (our design is four years old; much faster processors are now available).

In many speech recognition systems acoustic preprocessing encodes the speech signal into so-called LPC (linear predictive coding) coefficients,¹ which contain approximately the same information as the spectral decomposition. We preferred the FFT because, as will be shown, one of the main operations of the neural network that recognizes the phonemes is to perform metric clustering of the phonemic samples. The FFT, a transform of the signal, reflects its clustering properties better than a parametric code.

We had the option of applying the overall root-mean-square value of the speech signal as the extra sixteenth component in the pattern vector; in this way we expected to obtain more information on the transient signals. The recognition accuracy remained the same, however, within one percent. We believe that the acoustic processor can analyze many other speech features in addition to the spectral ones. Another trick that improved accuracy on the order of two percent was to make the true pattern vector out of two spectra 30 ms apart in the time scale. Since the two samples represent two different states of the signal, dynamic information is added

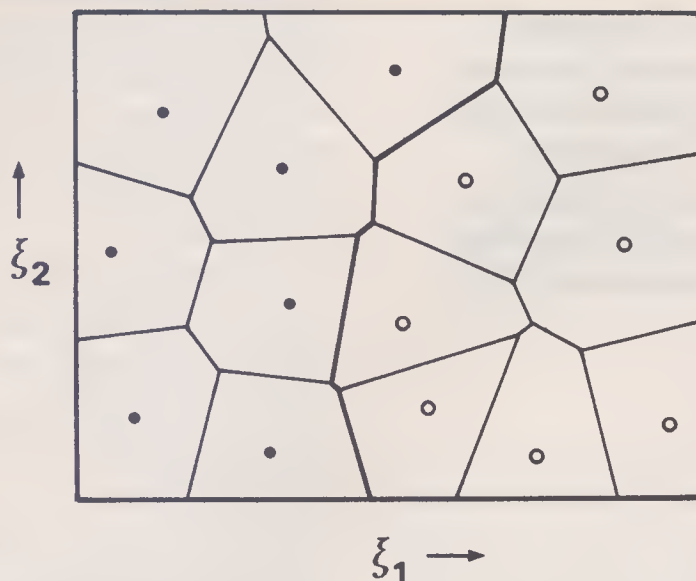


Figure 1. Voronoi tessellation partitions a two-dimensional (ξ_1 , ξ_2) "pattern space" into regions around reference vectors, shown as points in this coordinate system. All vectors (ξ_1 , ξ_2) in the same partition have the same reference vector as their nearest neighbor and are classified according to it. The solid and open circles, respectively, represent reference vectors of two classes, and the discrimination "surface" between them is drawn in bold.

to the preanalysis.

Because the plosives must be distinguished on the basis of the fast, transient parts of the speech waveform, we selected the spectral samples of the plosives from the transient regions of the signal, on the basis of the constancy of the waveform. On the other hand, there is evidence that the biological auditory system is sensitive not only to the spectral representations of speech but to their particular transient features too, and apparently it uses the nonlinear adaptive properties of the inner ear, especially its hair cells, the different transmission delays in the neural fibers, and many kinds of neural gating in the auditory nuclei (processing stations between the ear and the brain). For the time being, these nonlinear, dynamic neural functions are not understood well enough to warrant the design of standard electronic analogies for them.

Vector quantization

The instantaneous spectral power values on the 15 channels formed from the FFT

can be regarded as a 15-dimensional real vector in a Euclidean space. We might think that the spectra of the different phonemes of speech occupy different regions of this space, so that they could be detected by some kind of multidimensional discrimination method. In reality, several problems arise. One of them, as already stated, is that the distributions of the spectra of different phonemic classes overlap, so that it is not possible to distinguish the phonemes by any discrimination method with 100 percent certainty. The best we can do is to divide the space with optimal discrimination borders, relative to which, on the average, the rate of misclassifications is minimized. It turns out that analytical definition of such (nonlinear) borders is far from trivial, whereas neural networks can define them very effectively. Another problem is presented by the coarticulation effects discussed later.

A concept useful for the illustration of these so-called vector space methods for pattern recognition and neural networks is called *Voronoi tessellation*. For simplicity, consider that the dissimilarity of two or more spectra can be expressed in terms of

their vectorial difference (actually the norm of this difference) in an n -dimensional Euclidean space. Figure 1 exemplifies a two-dimensional space in which a finite number of *reference vectors* are shown as points, corresponding to their coordinates. This space is partitioned into regions, bordered by lines (in general, hyperplanes) such that each partition contains a reference vector that is the nearest neighbor to any vector within the same partition. These lines, or the midplanes of the neighboring reference vectors, constitute the Voronoi tessellation, which defines a set of *discrimination* or *decision surfaces*. This tessellation represents one kind of *vector quantization*, which generally means quantization of the vector space into discrete regions.

One or more neighboring reference vectors can be made to define a category in the vector space as the union of their respective partitions. Determination of such reference vectors was the main problem on which we concentrated in our neural network research. There are, of course, many classical mathematical approaches to this problem.³ In very simple and straightforward pattern recognition, samples, or prototypes, of earlier observed vectors are used as such for the reference vectors. For the new or unknown vector, a small number of its nearest prototypes are sought; then majority voting is applied to them to determine classification. A drawback of this method is that for good statistical accuracy an appreciable number of reference vectors are needed. Consequently, the comparison computations during classification, especially if they are made serially, become time-consuming; the unknown vector must be compared with all the reference vectors. Therefore, our aim was to describe the samples by a much smaller representative set of reference vectors without loss of accuracy.

Imagine now that a fixed number of discrete neurons is in parallel, looking at the speech spectrum, or the set of input signals. Imagine that each neuron has a template, a reference spectrum with respect to which the degree of matching with the input spectrum can be defined. Imagine further that the different neurons compete, the neuron with the highest matching score being regarded as the "winner." The input spectrum would then be assigned to the winner in the same way that an arbitrary vector is assigned to the closest reference vector and classified according to it in the above Voronoi tessellation.

There are neural networks in which such templates are formed adaptively, and which perform this comparison in parallel, so that the neuron whose template matches best with the input automatically gives an active response to it. Indeed, the self-organizing process described below defines reference vectors for the neurons such that their Voronoi tessellation sets near-optimal decision borders between the classes—i.e., the fraction of input vectors falling on the wrong side of the borders is minimized. In classical decision theory, theoretical minimization of the probability for misclassification is a standard procedure, and the mathematical setting for it is the Bayes theory of probability. In what follows, we shall thus point out that the vector quantization and nearest neighbor classification resulting in the neural network defines the reference vectors in such a way that their Voronoi tessellation very closely approximates the theoretical Bayesian decision surfaces.

The neural network

Detailed biophysical analysis of the phenomena taking place at the cell membrane of biological neurons leads to systems of nonlinear differential equations with dozens of state variables for each neuron; this would be untenable in a computational application. Obviously it is necessary to simplify the mathematics, while retaining some essentials of the real dynamic behavior. The approximations made here, while reasonably simple, are still rather “neural” and have been influential in many intriguing applications.

Figure 2 depicts one model neuron and defines its signal and state variables. The input signals are connected to the neuron with different, variable “transmittances” corresponding to the coupling strengths of the neural junctions called *synapses*. The latter are denoted by μ_{ij} (here i is the index of the neuron and j that of its input). Correspondingly, ξ_{ij} is the signal value (signal activity, actually the frequency of the neural impulses) at the j th input of the i th neuron.

Each neuron is thought to act as a pulse-frequency modulator, producing an output activity η_i (actually a train of neural impulses with this repetition frequency), which is obtained by integrating the input signals according to the following differential equation. (The biological neurons have an active membrane with a

capacitance that integrates input currents and triggers a volley of impulses when a critical level of depolarization is achieved.)

$$d\eta_i/dt = \sum_{j=1}^n \mu_{ij} \xi_{ij} - \gamma(\eta_i) \quad (1)$$

The first term on the right corresponds to the coupling of input signals to the neuron through the different transmittances; a linear, superpositive effect was assumed for simplicity. The last term, $-\gamma(\eta_i)$, stands for a nonlinear leakage effect that describes all nonideal properties, such as saturation, leakage, and shunting effects of the neuron, in a simple way. It is assumed to be a stronger than linear function of η_i . It is further assumed that the inverse function γ^{-1} exists. Then if the ξ_{ij} are held stationary, or they are changing slowly, we can consider the case $d\eta_i/dt \sim 0$, whereby the output will follow the integrated input as in a nonlinear, saturating amplifier according to

$$\eta_i = \sigma \left[\sum_{j=1}^n \mu_{ij} \xi_{ij} \right] \quad (2)$$

Here $\sigma[\cdot]$ is the inverse function of γ , and it usually has a typical sigmoidal form, with low and high saturation limits and a proportionality range between.

The settling of activity according to Equation 1 proceeds very quickly; in biological circuits it occurs in tens of milliseconds. Next we consider an adaptive process in which the transmittances μ_{ij} are assumed to change too. This is the effect regarded as “learning” in neural circuits, and its time constants are much longer. In biological circuits this process corresponds to changes in proteins and neural structures that typically take weeks. A simple, natural adaptation law that already has suggested many applications is the following: First, we must stipulate that parametric changes occur very selectively; thus dependence on the signals must be nonlinear. The classical choice made by most modelers is to assume that changes are proportional to the *product* of input and output activities (the so-called law of Hebb). However, this choice, as such, would be unnatural because the parameters would change in one direction only (notice that the signals are positive). Therefore it is necessary to modify this law—for example, by including some kind of nonlinear “forgetting” term. Thus we can write

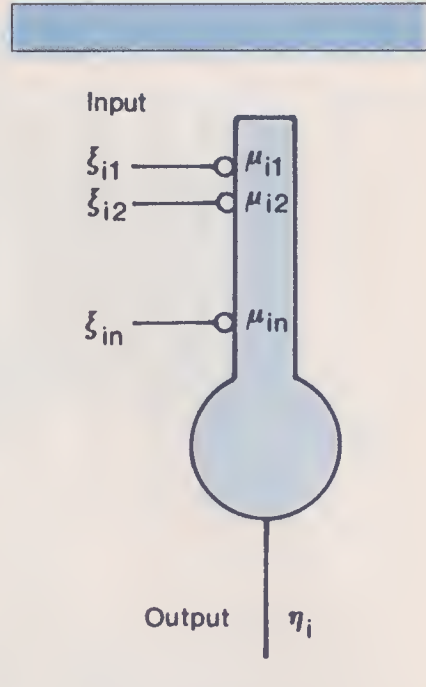


Figure 2. Symbol of a theoretical neuron and the signal and system variables relating to it. The small circles correspond to the input connections, the synapses.

$$d\mu_{ij}/dt = \alpha \eta_i \xi_{ij} - \beta(\eta_i) \mu_{ij} \quad (3)$$

where α is a positive constant, the first term is the “Hebbian” term, and the last term represents the nonlinear “forgetting” effect, which depends on the activity η_i ; forgetting is thus “active.” As will be pointed out later, the first term defines changes in the μ_{ij} in such a direction that the neuron tends to become more and more sensitive and selective to the particular combination of signals ξ_{ij} presented at the input. This is the basic adaptive effect.

On the other hand, to stabilize the output activity to a proper range, it seems very profitable for $\beta(\eta_i)$ to be a scalar function with a Taylor expansion in which the constant term is zero. Careful analyses have shown that this kind of neuron becomes selective to the so-called *largest principal component* of input.⁹ For many choices of the functional form, it can further be shown that the μ_{ij} will automatically become normalized such that the vector formed from the μ_{ij} during the process tends to a constant length (norm) indepen-

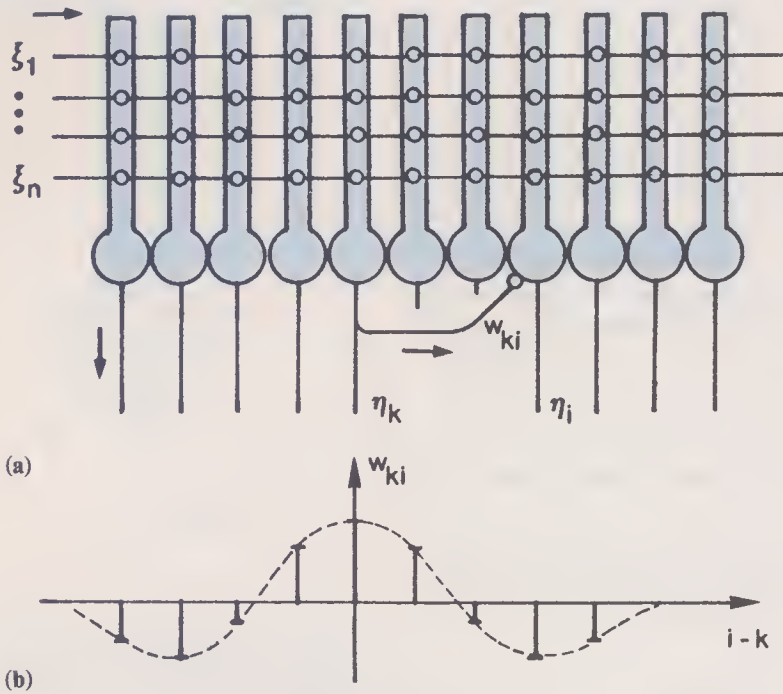


Figure 3. (a) Neural network underlying the formation of the phonotopic maps used in speech recognition. (b) The strengths of lateral interaction as a function of distance (the "Mexican hat" function).

dent of the signal values that occur in the process.⁹ We shall employ this effect a bit later in a further simplification of the model.

One cannot understand the essentials of neural circuits unless one considers their behavior as a *collective* system. An example occurs in the "self-organizing feature maps" in our speech recognition application. Consider Figure 3a, where a set of neurons forms a layer, and each neuron is connected to its neighbors in the lateral direction. We have drawn the network one-dimensionally for clarity, although in all practical applications it has been two-dimensional. The external inputs, in the simplest model used for pattern recognition, are connected in parallel to all the neurons of this network so that each neuron can simultaneously "look" at the same input. (Certain interesting but much more complex effects result if the input connections are made to different portions of the network and the activation is propagated through it in a sequence.)

The feedback connections are coupled to the neurons in the same way as the external inputs. However, for simplicity, only the latter are assumed to have adaptive synapses. If the feedbacks were adaptive, too, this network would exhibit other more complex effects.⁹ It should also be emphasized that the biological synaptic circuits of the feedbacks are different from those of the external inputs. The time-invariant coupling coefficient of the feedback connections, as a function of distance, has roughly the "Mexican hat" form depicted in Figure 3b, as in real neural networks. For negative coupling, signal-inverting elements are necessary; in biological circuits inversion is made by a special kind of inhibitory interneuron. If the external input is denoted

$$I_i = \sum_{j=1}^n \mu_{ij} \xi_{ij} \quad (4)$$

then the system equation for the network

activities η_i , denoting the feedback coupling from neuron k to neuron i by w_{ki} , can be written

$$d\eta_i/dt = I_i + \sum_{k \in S_i} w_{ki} \eta_k - \gamma(\eta_i) \quad (5)$$

where k runs over the subset S_i of those neurons that have connections with neuron i . A characteristic phenomenon, due to the lateral feedback interconnections, will be observed first: The initial activity distribution in the network may be more or less random, but over time the activity develops into clusters or "bubbles" of a certain dimension, as shown in Figures 4 and 5. If the interaction range is not much less than the diameter of the network, the network activity seems to develop into a single bubble, located around the maximum of the (smoothed) initial activity.

Consider now that there is no external source of activation other than that provided by the input signal connections, which extend in parallel over the whole network. According to Equations 1 and 2, the strength of the initial activation of a neuron is proportional to the dot product $m_i^T x$ where m_i is the vector of the μ_{ij} , x is the vector of the ξ_{ij} , and T is the transpose of a vector. (We use here concepts of matrix algebra whereby m_i and x are column vectors.) Therefore, the bubble is formed around those units at which $m_i^T x$ is maximum.

The saturation limits of $\sigma[\cdot]$ defined by Equation 2 stabilize the activities η_i to either a low or a high value. Similarly, $\beta(\eta_i)$ takes on either of two values. Without loss of generality, it is possible to rescale the variables ξ_{ij} and μ_{ij} to make $\eta_i \in \{0, 1\}$, $\beta(\eta_i) \in \{0, \alpha\}$, whereby Equation 3 will be further simplified and split in two equations:

$$d\mu_{ij}/dt = \alpha(\xi_{ij} - \mu_{ij}) \quad (6a) \\ \text{if } \eta_i = 1 \text{ and } \beta = \alpha \text{ (inside the bubble)}$$

$$d\mu_{ij}/dt = 0 \quad (6b) \\ \text{for } \eta_i = \beta = 0 \text{ (outside the bubble)}$$

It is evident from Equation 6 that the transmittances μ_{ij} then adaptively tend to follow up the input signals ξ_{ij} . In other words, these neurons start to become selectively sensitized to the prevailing input pattern. But this occurs only when the bubble lies over the particular neuron. For another input, the bubble lies over other neurons, which then become sensitized to that input. In this way different parts of

the network are automatically "tuned" to different inputs.

The network will indeed be tuned to different inputs in an ordered fashion, as if a continuous map of the signal space were formed over the network. The continuity of this mapping follows from the simple fact that the vectors m_i of contiguous units (within the bubbles) are modified in the same direction, so that during the course of the process the neighboring values become smoothed. The ordering of these values, however, is a very subtle phenomenon, the proof or complete explanation of which is mathematically very sophisticated⁹ and cannot be given here. The effect is difficult to visualize without, say, an animation film. A concrete example of this kind of ordering is the phonotopic map described later in this article.

Shortcut learning algorithm

In the time-continuous process just described, the weight vectors attain asymptotic values, which then define a vector quantization of the input signal space, and thus a classification of all its vectors. In practice, the same vector quantization can be computed much more quickly from a numerically simpler algorithm. The bubble is equivalent to a neighborhood set N_c of all those network units that lie within a certain radius from a certain unit c . It can be shown that the size of the bubble depends on the interaction parameters, and so we can reason that the radius of the bubble is controllable, eventually being definable as some function of time. For good self-organizing results, it has been found empirically that the radius indeed should decrease in time monotonically. Similarly $\alpha = \alpha(t)$ ought to be a monotonically decreasing function of time. Simple but effective choices for these functions have been determined in a series of practical experiments.⁹

As stated earlier, the process defined by Equation 1 normalizes the weight vectors m_i to the same length. Since the bubble is formed around those units at which $m_i^T x$ is maximum, its center also coincides with that unit for which the norm of the vectorial difference $x - m_i$ is minimum.

Combining all the above results, we obtain the following shortcut algorithm. Let us start with random initial values $m_i = m_i(0)$. For $t = 0, 1, 2, \dots$, compute:

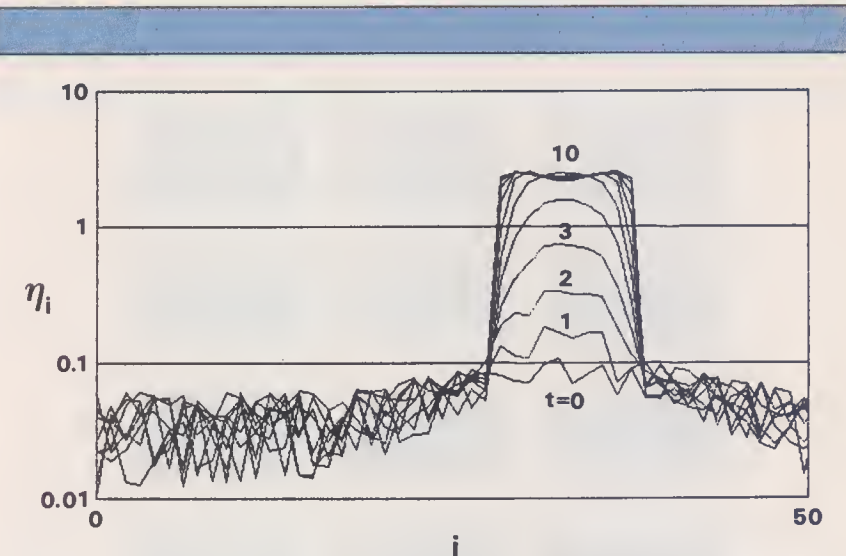


Figure 4. Development of the distribution of activity over time (t) into a stable "bubble" in a laterally interconnected neural network (cf. Figure 3). The activities of the individual neurons (η_i) are shown in the logarithmic scale.

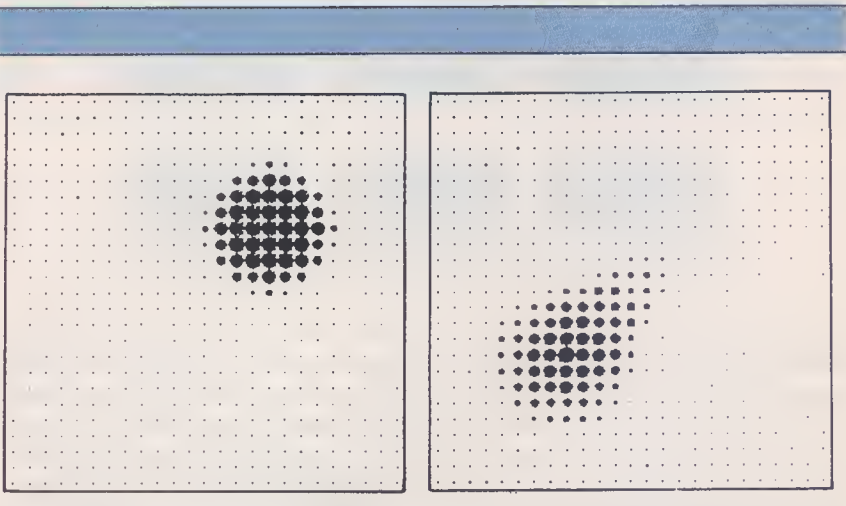


Figure 5. "Bubbles" formed in a two-dimensional network viewed from the top. The dots correspond to neurons, and their sizes correspond to their activity. In the picture on the right, the input was changing slowly, and the motion of the bubble is indicated by its "tail."

(1) Center of the bubble (c):

$$\|x(t) - m_c(t)\| = \min_i \{\|x(t) - m_i(t)\|\} \quad (7a)$$

(2) Updated weight vectors:

$$m_i(t+1) = m_i(t) + \alpha(t) (x(t) - m_i(t)) \quad \text{for } i \in N_c$$

$$m_i(t+1) = m_i(t) \quad \text{for all other indices } i \quad (7b)$$

As stated above, $\alpha = \alpha(t)$ and $N_c = N_c(t)$ are empirical functions of time. The asymptotic values of the m_i define the vector quantization. Notice, too, that Equation 7a defines the classification of input according to the closest weight vector to x .

We must point out that if N_c contained the index i only, Equations 7a and 7b

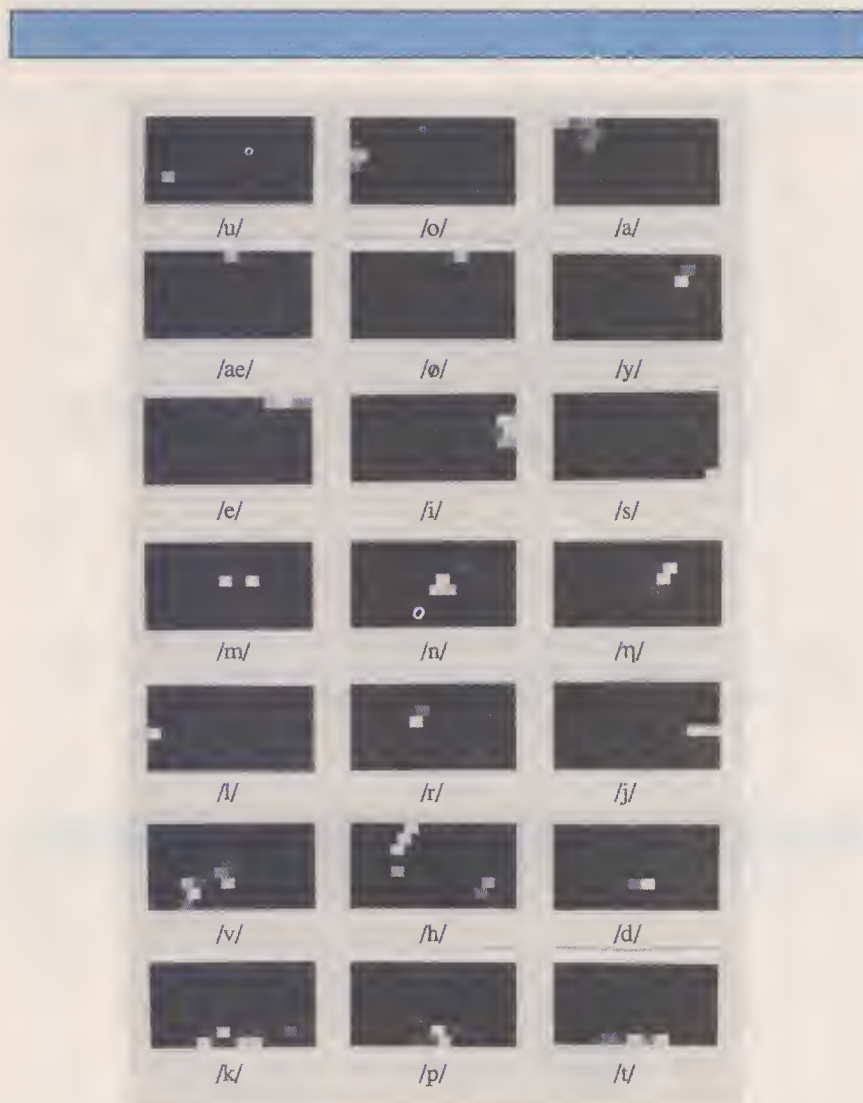


Figure 6. The signal of natural speech is preanalyzed and represented on 15 spectral channels ranging from 200 Hz to 5 kHz. The spectral powers of the different channel outputs are presented as input to an artificial neural network. The neurons are tuned automatically, without any supervision or extra information, to the acoustic units of speech identifiable as phonemes. In this set of pictures the neurons correspond to the small rectangular subareas. Calibration of the map was made with 50 samples of each test phoneme. The shaded areas correspond to histograms of responses from the map to certain phonemes (white: maximum).

would superficially resemble the classical vector quantization method called *k-means clustering*.¹⁰ The present method, however, is more general because the corrections are made over a wider, dynamically defined neighborhood set, or bubble N_c , so that an *ordered* mapping is obtained. Together with some fine adjustments of the m_i vectors,⁹ spectral recognition accuracy is improved significantly.

Phonotopic maps

For this discussion we assume that a lattice of hexagonally arranged neurons forms a two-dimensional neural network of the type depicted in Figure 3. As already described, the microphone signal is first converted into a spectral representation, grouped into 15 channels. These channels together constitute the 15-component

stochastic input vector x , a function of time, to the network. The self-organizing process has been used to create a "topographic," two-dimensional map of speech elements onto the network.

Superficially this network seems to have only one layer of neurons; due to the lateral interactions in the network, however, its topology is in effect even more complicated than that of the famous multilayered Boltzmann machines or backpropagation networks.¹¹ Any neuron in our network is also able to create an internal representation of input information in the same way as the "hidden units" in the backpropagation networks eventually do. Several projects have recently been launched to apply Boltzmann machines to speech recognition. We should learn in the near future how they compete with the design described here.

The input vectors x , representing short-time spectra of the speech waveform, are computed in our system every 9.83 milliseconds. These samples are applied in Equations 7a and 7b as input data in their natural order, and the self-organizing process then defines the m_i , or the weight vectors of the neurons. One striking result is that the various neurons of the network become sensitized to spectra of different phonemes and their variations in a two-dimensional order, although teaching was not done by the phonemes; only spectral samples of input were applied. The reason is that the input spectra are clustered around phonemes, and the process finds these clusters. The maps can be calibrated using spectra of known phonemes. If then a new or unknown spectrum is presented at the inputs, the neuron with the closest transmittance vector m_i gives the response, and so the classification occurs in accordance with the Voronoi tessellation in which the m_i act as reference vectors. The values of these vectors very closely reflect the actual speech signal statistics.¹¹ Figure 6 shows the calibration result for different phonemic samples as a gray-level histogram of such responses, and Figure 7 shows the map when its neurons are labeled according to the majority voting for a number of different responses.

The speech signal is a continuous waveform that makes transitions between various states, corresponding to the phonemes. On the other hand, as stated earlier, the plosives are detectable only as transient states of the speech waveform. For that reason their labeling in Figure 7 is not reliable. Recently we solved the

problem of more accurate detection of plosives and certain other phonemic categories by using special, auxiliary maps in which only a certain category of phonemes was represented, and which were trained by a subset of samples. For this purpose we first detect the presence of such phonemes (as a group) from the waveform, and then we use this information to activate the corresponding map. For instance, the occurrence of /k,p,t/ is indicated by low signal energy, and the corresponding spectral samples are picked from the transient regions following silence. The nasals as a group are detectable by responses obtained from the middle area of the main map.

Another problem is *segmentation* of the responses from the map into a standard phonemic transcription. Consider that the spectral samples are taken at regular intervals every 9.83 milliseconds, and they are first labeled in accordance with the corresponding phonemic spectra. These labeled samples are called *quasiphonemes*; in contrast, the duration of a true phoneme is variable, say, from 40 to 400 milliseconds. We have used several alternative rules for the segmentation of quasiphoneme sequences into true phonemes. One of them is based on the degree of stability of the waveform; most phonemes, let alone plosives, have a unique stationary state. Another, more heuristic method is to decide that if m out of n successive quasiphonemes are the same, they correspond to a single phoneme; e.g., $m=4$ and $n=7$ are typical values.

The sequences of quasiphonemes can also be visualized as trajectories over the main map, as shown in Figure 8. Each arrowhead represents one spectral sample. For clarity, the sequence of coordinates shown by arrows has been slightly smoothed to make the curves more continuous. It is clearly discernible that convergence points of the speech waveform seem to correspond to certain (stationary) phonemes.

This kind of graph provides a new means, in addition to some earlier ones, for the visualization of the phonemes of speech, which may be useful for speech training and therapy. Profoundly deaf people may find it advantageous to have an immediate visual feedback from their speech.

It may be necessary to point out that the phonotopic map is not the same thing as the so-called formant maps used in phonetics. The latter display the speech signal in coordinates that correspond to the two lowest formants, or resonant frequencies



Figure 7. The neurons, shown as circles, are labeled with the symbols of the phonemes to which they "learned" to give best responses. Most neurons give a unique answer; the double labels here show neurons that respond to two phonemes. Distinction of /k,p,t/ from this map is not reliable and needs the analysis of the transient spectra of these phonemes by an auxiliary map. In the Japanese version there are auxiliary maps for /k,p,t/, /b,d,g/, and /m,n,ŋ/ for more accurate analysis.

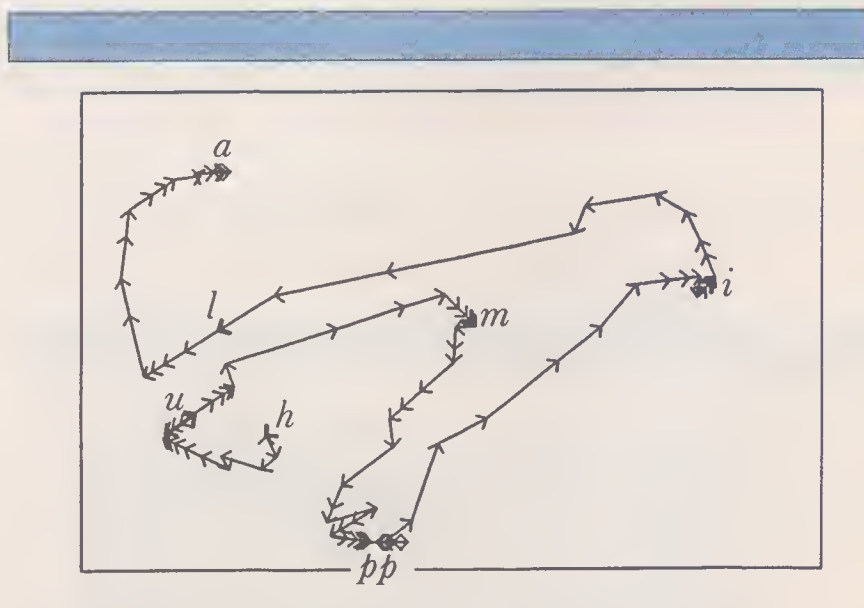


Figure 8. Sequence of the responses obtained from the phonotopic map when the Finnish word *humpilla* was uttered. The arrows correspond to intervals of 9.83 milliseconds, at which the speech waveform was analyzed spectrally.

of the vocal tract. Neither is this map any kind of principal component graph for phonemes. The phonotopic map displays the images of the complete spectra as points on a plane, the distances of which

approximately correspond to the *vectorial differences* between the original spectra; so this map should rather be regarded as a *similarity graph*, the coordinates of which have no explicit interpretation.

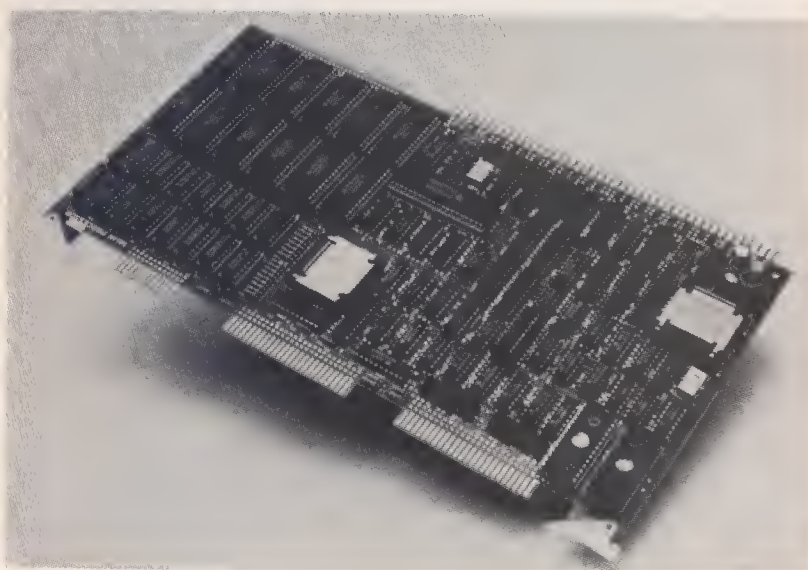


Figure 9. The coprocessor board for the neural network and the postprocessing functions.

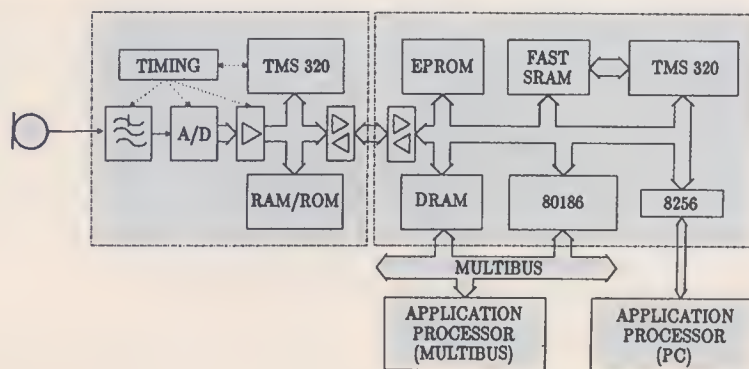


Figure 10. Block diagram of the coprocessor board. A/D: analog-to-digital converter. TMS320: Texas Instruments 32010 signal processor chip. RAM/ROM: 4K-word random-access memory, 256-word programmable read-only memory. EPROM: 64K-byte electrically erasable read-only memory. DRAM: 512K-byte dual-port random-access memory. SRAM: 96K-byte paged dual-port random-access memory. 80186: Intel microprocessor CPU. 8256: parallel interface.

Actually, the phoneme recognition accuracy can still be improved by three or four percent if the templates m_i are fine-tuned: small corrections to the responding neurons can be made automatically by turning their template vectors toward x if a tentative classification was correct, and away from x if the result was wrong.

Postprocessing in symbolic form

Even if the classification of speech spectra were error-free, the phonemes would not be identifiable from them with 100-percent reliability. This is because there are *coarticulation effects* in speech: the phonemes are influenced by neighboring phonemes. One might imagine it possible to list and take into account all such variations. But there may be many hundreds of different *frames* or *contexts* of neighboring phonemes in which a particular phoneme may occur. Even this, however, is an optimistic figure since the neighbors too may be transformed by other coarticulation effects and errors. Thus, the correction of such transformed phonemes should be made by reference to some kind of *context-sensitive stochastic grammar*, the rules of which are derived from real examples. I have developed a program code that automatically constructs the grammatical transformation rules on the basis of speech samples and their correct reference transcriptions.¹² A typical error grammar may contain 15,000 to 20,000 rules (productions), and these rules can be encoded as a data structure or stored in an associative memory. The optimal amount of context is determined automatically for each rule separately. No special hardware is needed; the search of the matching rules and their application can be made in real time by efficient and fast software methods, based on so-called hash coding, without slowing down the recognition operation.

The two-stage speech recognition system described in this article is a genuine phonetic typewriter, since it outputs orthographic transcriptions for unrestricted utterances, the forms of which only approximately obey certain morphological rules or regularities of a particular language. We have implemented this system for both Finnish and (romanized) Japanese. Both of these languages, like Latin, are characterized by the fact that their orthography is almost identical to their phonemic transcription.

As a complete speech recognition device, our system can be made to operate in either of two modes: (1) transcribing dictation of *unlimited* text, whereby the words (at least in some common idioms) can be connected, since similar rules are applicable for the editing of spaces between the words (at present short pauses are needed to insert spaces); and (2) isolated-word recognition from a large vocabulary.

In isolated-word recognition we first use the phonotopic map and its segmentation algorithm to produce a raw phonemic transcription of the uttered word. Then this transcription is compared with reference transcriptions earlier collected from a great many words. Comparison of partly erroneous symbolic expressions (strings) can be related to many standard similarity criteria. Rapid prescreening and spotting of the closest candidates can again be performed by associative or hash-coding methods; we have introduced a very effective error-tolerant searching scheme called *redundant hash addressing*, by which a small number of the best candidates, selected from vocabularies of thousands of items, can be located in a few hundred milliseconds (on a personal computer). After that, the more accurate final comparison between the much smaller number of candidates can be made by the best statistical methods.

Hardware implementations and performance

The system's neural network could, in principle, be built up of parallel hardware components that behave according to Equations 5 and 6. For the time being, no such components have been developed. On the other hand, for many applications the equivalent functions from Equations 7a and 7b are readily computable by fast digital signal processor chips; in that case the various neurons only exist *virtually*, as the signal processors are able to solve their equations by a timesharing principle. Even this operation, however, can be performed in real time, especially in speech processing.

The most central neural hardware of our system is contained on the coprocessor board shown in Figure 9. Its block diagram is shown in Figure 10. Only two signal processors have been necessary: one for the acoustic preprocessor that

Neural Networks Begin with MIT/Bradford Books

NEURAL NETWORKS AND NATURAL INTELLIGENCE

Stephen Grossberg

This anthology of the latest research in neural networks is packed with real-time computer simulations and rigorous demonstrations, covering results in vision, speech, cognitive information processing, adaptive pattern recognition, adaptive robotics, conditioning and attention, cognitive-emotional interactions, and decision making under risk.

\$29.95

NEUROCOMPUTING

Foundations of Research

edited by James A. Anderson and Edward Rosenfeld

A fundamental reference work that collects seminal work by McCulloch and Pitts, Hebb, Lashley, von Neumann, Minsky and Papert, Cooper, Grossberg, Kohonen, and McClelland and Rumelhart.

\$55.00

NATURAL COMPUTATION

Selected Readings

edited by Whitman A. Richards

This extensive book of readings combines mathematics, artificial intelligence, computer science, experimental psychology, and neurophysiology in studying perception.

\$25.00 paper (\$50.00 cloth)

PARALLEL DISTRIBUTED PROCESSING

Explorations in the Microstructure of Cognition

Volume 1: Foundations

David E. Rumelhart, James L. McClelland, and the PDP Research Group

Volume 2: Psychological and Biological Models

James L. McClelland, David E. Rumelhart, and the PDP Research Group

\$16.95 each volume, paper

\$29.95 the set

EXPLORATIONS IN PARALLEL DISTRIBUTED PROCESSING

A Handbook of Models, Programs, and Exercises

James L. McClelland and David E. Rumelhart

\$27.50 paper, software included for IBM PC



Two MIT classics now available in paperback

PERCEPTRONS

Expanded Edition

Marvin L. Minsky and Seymour Papert

"The best place to begin reviewing neural networks is the late 1960s. In their landmark book, Minsky and Papert examined the notion of building 'thinking machines' by joining together computational units that mimic human neurons."—*IEEE Expert*

\$12.50 paper

EMBODIMENTS OF MIND

Warren S. McCulloch

Preface by Jerome Y. Lettvin

Another major work of the 1960s that teems with concepts that are highly relevant to current developments in neuroscience and neural networks.

\$12.50 paper

To order call toll free: 800-356-0343 or (617) 253-2884.
MasterCard and Visa accepted.

The MIT Press

55 Hayward Street, Cambridge, MA 02142

produces the input pattern vectors x , and another for timeshared computation of the responses from the neural network. For the time being, the self-organized computation of the templates m_i , or "learning," is made in an IBM PC AT-compatible host processor, and the transmittance parameters (synaptic transmittances) are loaded onto the coprocessor board. Newer designs are intended to operate as stand-alone systems. A standard microprocessor CPU chip on our board takes care of overall control and data routing and performs some preprocessing operations after FFT (such as logarithmization and normalization), as well as segmenting the quasiphoneme strings and deciding whether the auxiliary transient maps are to be used. Although the 80186 is a not-so-effective CPU, it still has extra capacity for postprocessing operations: it can be programmed to apply the context-sensitive grammar for unlimited text or to perform the isolated-word recognition operations.

The personal computer has been used during experimentation for all postprocessing operations. Nonetheless, the overall recognition operations take place in near real time. In the intended mode of operation the speech recognizer will only assist the keyboard operations and communicate with the CPU through the same channel.

One of the most serious problems with this system, as well as with any existing speech recognizer, is recognition accuracy, especially for an arbitrary speaker. After postprocessing, the present transcription accuracy varies between 92 and 97 percent, depending on speaker and difficulty of text. We performed most of the experiments reported here with half a dozen male speakers, using office text, names, and the most frequent words of the language. The number of tests performed over the years is inestimable. Typically, thousands of words have been involved in a particular series of tests. Enrollment of a new speaker requires dictation of 100 words, and the learning processes can proceed concurrently with dictation. The total learning time on the PC is less than 10 minutes. During learning, the template vectors of the phonotopic map are tuned to the new samples.

Isolated-word recognition from a 1000-word vocabulary is possible with an accuracy of 96 to 98 percent. Since the recognition system forms an intermediate symbolic transcription that can be compared with any standard reference tran-

scriptions, the vocabulary or its active subsets can be defined in written form and changed dynamically during use, without the need of speaking any samples of these words.

All output, for unlimited text as well as for isolated words, is produced in near real time: the mean delay is on the order of 250 milliseconds per word. It should be noticed that contemporary microprocessors already have much higher speeds (typically five times higher) than the chips used in our design.

To the best of our knowledge, this system is the only existing complete speech recognizer that employs neural computing principles and has been brought to a commercial stage, verified by extensive tests. Of course, it still falls somewhat short of expectations; obviously some kind of linguistic postprocessing model would improve its performance. On the other hand, our principal aim was to demonstrate the highly adaptive properties of neural networks, which allow a very accurate, nonlinear statistical analysis of real signals. These properties ought to be a goal of all practical "neurocomputers." □

References

1. W.A. Lea, ed., *Trends in Speech Recognition*, Prentice-Hall, Englewood Cliffs, N.J., 1980.
2. S.E. Levinson, L.R. Rabiner, and M.M. Sondhi, "An Introduction to the Application of the Theory of Probabilistic Functions of a Markov Process to Automatic Speech Recognition," *Bell Syst. Tech. J.*, Apr. 1983, pp. 1035-1073.
3. P.A. Devijver and J. Kittler, *Pattern Recognition: A Statistical Approach*, Prentice-Hall, London, 1982.
4. D.H. Klatt, "Review of the ARPA Speech Understanding Project," *J. Acoust. Soc. Amer.*, Dec. 1977, pp. 1345-1366.
5. R. Reddy and V. Zue, "Recognizing Continuous Speech Remains an Elusive Goal," *IEEE Spectrum*, Nov. 1983, pp. 84-87.
6. P. Petre, "Speak, Master: Typewriters That Take Dictation," *Fortune*, Jan. 7, 1985, pp. 56-60.
7. M.R. Schroeder and J.L. Hall, "Model for Mechanical to Neural Transduction in the Auditory Receptor," *J. Acoust. Soc. Am.*, May 1974, pp. 1055-1060.
8. R. Meddis, "Simulation of Mechanical to Neural Transduction in the Auditory Receptor," *J. Acoust. Soc. Am.*, Mar. 1986, pp. 703-711.

9. T. Kohonen, *Self-Organization and Associative Memory*, Series in Information Sciences, Vol. 8, Springer-Verlag, Berlin-Heidelberg-New York-Tokyo, 1984; 2nd ed. 1988.
10. J. Makhoul, S. Roucos, and H. Gish, "Vector Quantization in Speech Coding," *Proc. IEEE*, Nov. 1985, pp. 1551-1588.
11. D.E. Rumelhart, G.E. Hinton, and R.J. Williams, "Learning Internal Representations by Error Propagation," in *Parallel Distributed Processing, Explorations in the Microstructure of Cognition, Volume 1: Foundations*, ed. by David E. Rumelhart, James L. McClelland, and the PDP Research Group, MIT Press, Cambridge, Mass., 1986, pp. 318-362.
12. T. Kohonen, "Dynamically Expanding Context, with Application to the Correction of Symbol Strings in the Recognition of Continuous Speech," *Proc. Eighth Int'l Conf. Pattern Recognition*, IEEE Computer Society, Washington, D.C., 1986, pp. 1148-1151.



Teuvo Kohonen is a professor on the Faculty of Information Sciences of Helsinki University of Technology, Finland. He is also a research professor of the Academy of Finland, a member of the Finnish Academy of Sciences, and a member of the Finnish Academy of Engineering Sciences. He received his D.Eng. degree in physics from Helsinki University of Technology in 1962. His scientific interests are neural computers and pattern recognition.

Kohonen has written four textbooks, of which *Content-Addressable Memories* (Springer, 1987) and *Self-Organization and Associative Memory* (Springer, 1988) are best known. He was the first vice chairman of the International Neural Network Society. Kohonen is a senior member of the IEEE. He was awarded the Eemil Aaltonen Honorary Prize in 1983 and the Cultural Prize of Finnish Commercial Television in 1984.

Kohonen's address is Helsinki University of Technology, Laboratory of Computer and Information Science, Rakentajanaukio 2 C, SF-02150 Espoo, Finland.

PREPARE FOR TAKE-OFF

Meta Software launches Design/IDEF™, the first CIM modeling tool for desktop engineering.

Design/IDEF is the fast, accurate and cost-effective way to automate Computer Integrated Manufacturing (CIM) planning and modeling.

Now you can create, read and comment on CIM models—quickly and easily. And make any modifications instantly. So projects are completed in less time. At a substantial savings.

Design/IDEF is a smart, high-performance graphics tool that's powerful enough to handle CIM models containing hundreds of hierarchically linked diagrams. It supports the CIM systems design methodology used by the US Air Force, major aerospace contractors, and large manufacturing companies.



No more re-drawing, re-drawing, re-drawing.

If you move or resize an object, Design/IDEF automatically recreates all associated arrows, text and subordinate objects—without affecting arrow angles or curvature. Since text and graphics are combined, you can easily associate text with boxes or arrows.

As your CIM model evolves, you can move detail to a subpage. Design/IDEF will automatically maintain the relationships and display the hierarchy. So you save time and improve accuracy.

What's more, an interactive data dictionary provides centralized storage, organization, and reporting. And a data capture facility makes it easy to accurately monitor and update files.

Special Introductory Offer.

Design/IDEF is priced at \$2500, but you can purchase it for \$2000 until March 31, 1988. So act now and discover why so many top manufacturing and engineering companies use Meta Software's Design/IDEF to speed up projects and keep costs under control.

Write or call toll-free 1-800-227-4106 for more information about Design/IDEF. Inside Massachusetts, call 617-576-6920. And fly through CIM modeling in record time.

Available for the Macintosh™ Plus, SE, II. Macintosh is a trademark of Apple Computer, Inc. Design/IDEF is a trademark of META Software Corporation.

Meta Software



Meta Software Corporation,
150 CambridgePark Drive,
Cambridge, Massachusetts 02140.

Reader Service Number 3

IBM PC-BASED

ANALYST WORKBENCH

- Modular
- Flexible
- Affordable

Your pipeline to productivity in software development, design and documentation

BUILD SYSTEMS FASTER

Priced for
your productivity
One for every analyst

\$595.00 per tool
quantity discounts

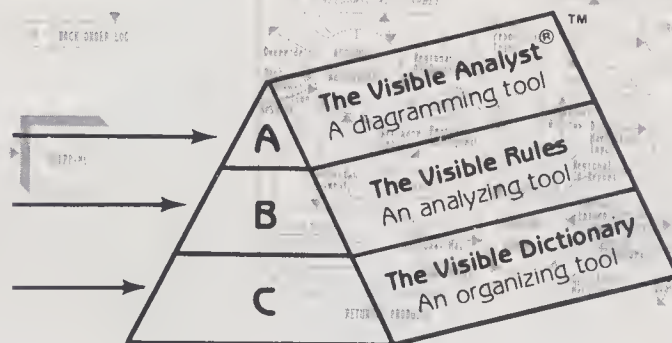
Featuring —

THE VISIBLE ANALYST® WORKBENCH

Design Tool A

Design Tool B

Design Tool C



Software Tools for Software Professionals

FEATURES:

1. Mouse-driven graphics allow analyst to build and test models quickly and accurately. Graphics package includes reuseable custom symbols, reuseable constructs, sizing and stylizing of symbols, variety of line styles and terminators, variety of character sets including bold, italics and sideways. Text sizable for large titles and labels, intelligent graphics object oriented editing, nesting, paging, export/import, 180 dpi high quality output...and more. Designed to handle all of the leading structured methodologies as well as diagramming conventions that require special symbols or configurations.
2. Expert Rules System allows for menu selection of leading structured methodologies. Yourdon, Gane and Sarson supported. Specialized or Custom Rules can also be implemented. Validates and balances data flow diagrams over any number of levels...and more.
3. Key information automatically loaded into data dictionary. This information can describe processes, data elements, data records, various files, charts, narratives, objects, etc. Provides a cross index, global or wild card search, replace and delete items, checks for aliases, does audit trail and produces reports in various formats...and more.
4. Low cost provides you with standards throughout your department or company - better communications means better systems faster...start mapping your information today.

For more information contact your nearest dealer or distributor or call, write, or telex.

CALL 617-969-4100 or Telex 261102

Start Experiencing THE VISIBLE SOLUTION™ Today!!!!

VISIBLE SYSTEMS CORPORATION

49 Lexington St., Newton, MA 02165

© 1988 Visible Systems Corporation Visible, The Visible Analyst and Visible Systems Corporation are registered trademarks of Visible Systems Corporation



Neural Nets for Adaptive Filtering and Adaptive Pattern Recognition

Bernard Widrow, Stanford University

Rodney Winter, United States Air Force

The fields of adaptive signal processing and adaptive neural networks have been developing independently but have the adaptive linear combiner (ALC) in common. With its inputs connected to a tapped delay line, the ALC becomes a key component of an adaptive filter. With its output connected to a quantizer, the ALC becomes an adaptive threshold element or adaptive neuron.

Adaptive filters have enjoyed great commercial success in the signal processing field. All high-speed modems now use adaptive equalization filters. Long-distance telephone and satellite communications links are being equipped with adaptive echo cancelers to filter out echo, allowing simultaneous two-way communications. Other applications include noise canceling and signal prediction.

Adaptive threshold elements, on the other hand, are the building blocks of neural networks. Today neural nets are the focus of widespread research interest. Areas of investigation include pattern recognition and trainable logic. Neural network systems have not yet had the commercial impact of adaptive filtering.

The commonality of the ALC to adaptive signal processing and adaptive neural networks suggests the two fields have much to share with each other. This article describes practical applications of the ALC in signal processing and pattern recognition.



A new multilayer adaptation algorithm that descrambles output and reproduces original patterns is advancing the practicality of neural-network pattern-recognition systems.

The adaptive linear combiner

The ALC shown in Figure 1 is the basic building block for most adaptive systems. The output is a linear combination of the many input signals. The weighting coefficients comprise a weight vector. The input signals comprise an input signal vector. The output signal is the inner product or dot product of the input signal vector with

the weight vector. The output signal is compared with a special input signal called the desired response, and the difference is the error signal. To optimize performance, the ALC's weighting coefficients or weights are generally adjusted to minimize the mean square of the error signal. Of the many adaptive algorithms to adjust the weights automatically, the most popular is the Widrow-Hoff LMS (least mean square) algorithm devised in 1959.¹

Adaptive filters. Digital signals generally originate from sampling continuous input signals by analog-to-digital conversion. Digital signals are often filtered by means of a tapped delay line or transversal filter, as shown in Figure 2a. The sampled input signal is applied to a string of delay boxes, each delaying the signal by one sampling period. An ALC is seen connected to the taps between the delay boxes. The filtered output is a linear combination of the current and past input signal samples. By varying the weights, the impulse response from input to output is directly controllable. Since the frequency response is the Fourier transform of the impulse response, controlling the impulse response controls the frequency response. The weights are usually adjusted so that the output signal will provide the best least-squares match over time to the desired-response input signal.

The literature reports many other forms

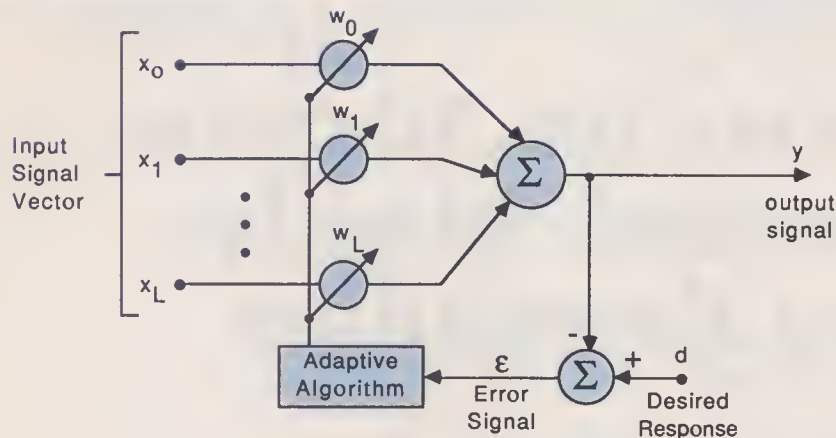


Figure 1. Adaptive linear combiner (ALC).

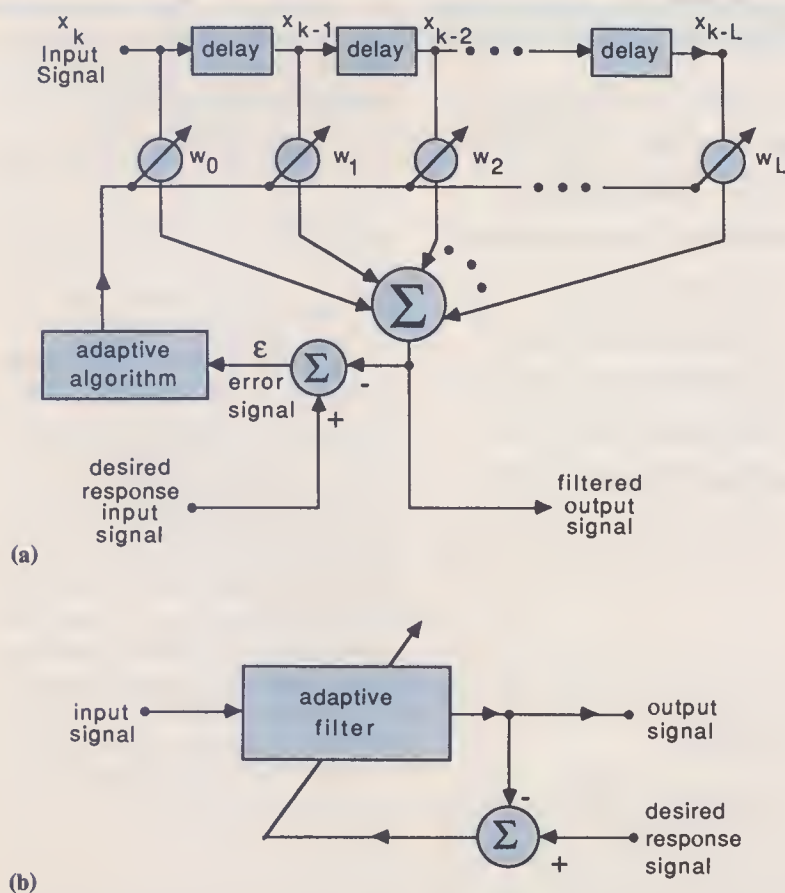


Figure 2. Adaptive digital filter: (a) details of a tapped-delay-line digital filter; (b) symbolic representation of an adaptive filter.

of adaptive filters.¹ Some use feedback to obtain both poles and zeros. The filter of Figure 2a realizes only zeros. Adaptive filters based on lattice structures achieve more rapid convergence under certain conditions. However, the simplest, most robust, and most widely used filter is that of Figure 2a adapted by the LMS algorithm. Figure 2b shows a symbolic representation of an adaptive filter.

Adaptive threshold element. Figure 3 shows an adaptive threshold element, a key component in adaptive pattern recognition systems. It consists of an adaptive linear combiner cascaded with a quantizer. The output of the ALC is quantized to produce a binary "decision." Most often, the inputs are binary and the desired response is binary. As such, the adaptive threshold element is trainable and capable of implementing binary logic functions. The LMS algorithm was originally developed to train the adaptive threshold element of Figure 3. This element was called an adaptive linear neuron or Adaline.²

The adaptive threshold element was an early neuronal model. The adaptive weights were analogous to synapses. The input vector components related to the dendritic inputs. The quantized output was analogous to the axonal output. The output decision was determined by a weighted sum of the inputs, in much the same way real neurons were believed to behave.

Adaptive signal processing

The adaptive filter of Figure 2b has an input signal and produces an output signal. The desired response is supplied during training. A question naturally arises: If the desired response were known and available, why would one need the adaptive filter? Put another way, how would one obtain the desired response in a practical application? There is no general answer to these questions, but studying successful examples provides some insight.

Example 1—system modeling. In many engineering and scientific applications, a system of unknown structure has observable input and output signals. One way of obtaining knowledge about the unknown system's dynamic response is to apply its input to an adaptive filter and to use its output as the adaptive filter's desired response. (See Figure 4.) The adaptive fil-

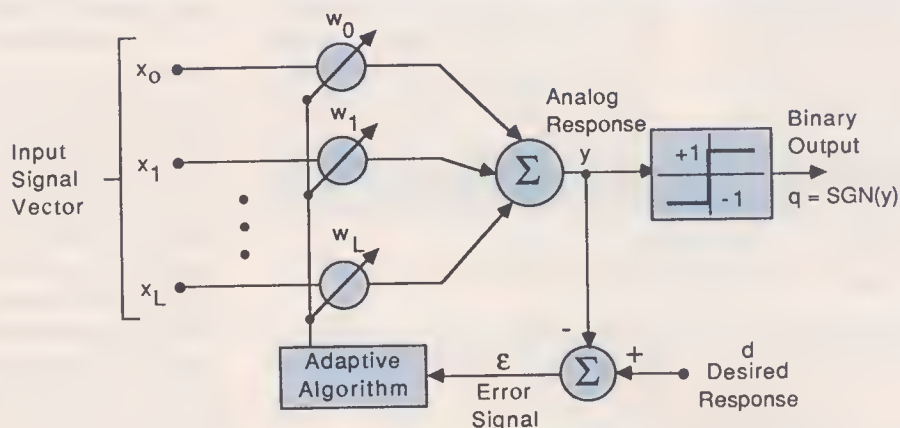


Figure 3. Adaptive threshold element (Adaline).

ter develops an impulse response to match that of the unknown system since the filter and the system develop similar outputs when driven by the same input.

Example 2—statistical prediction. One can estimate future values of time-correlated digital signals from present and past input samples. Wiener³ has developed optimal linear least-squares filtering techniques for signal prediction. When the signal's autocorrelation function is known, Wiener's theory yields the impulse response of the optimal filter. More often than not, however, the autocorrelation function is unknown and may be time-variable. One could use a correlator to measure the autocorrelation function and plug this into Wiener theory to get the optimal impulse response, or one could get the optimal prediction filter directly by adaptive filtering. Figure 5 illustrates the latter approach.

In this figure, the input signal delayed by Δ time units is fed to an adaptive filter. The undelayed input serves as the desired response for this adaptive filter. The filter weights adapt and converge to produce a best least-squares estimate of the present input signal, given an input that is this very signal delayed by Δ . The optimal weights are copied into a "slave filter" whose input is undelayed and whose output therefore is a best least-squares prediction of the input Δ time units into the future.

Example 3—noise canceling. Separating a signal from additive noise is a common

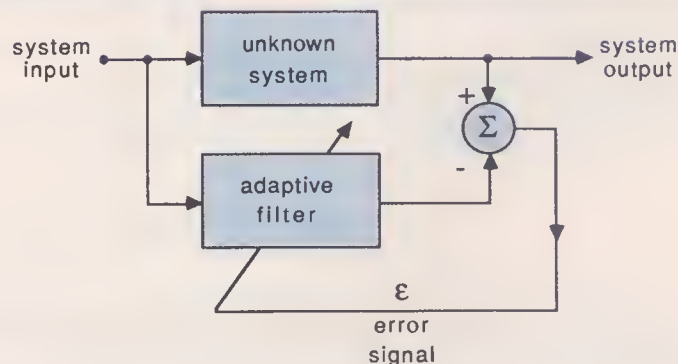


Figure 4. System modeling.

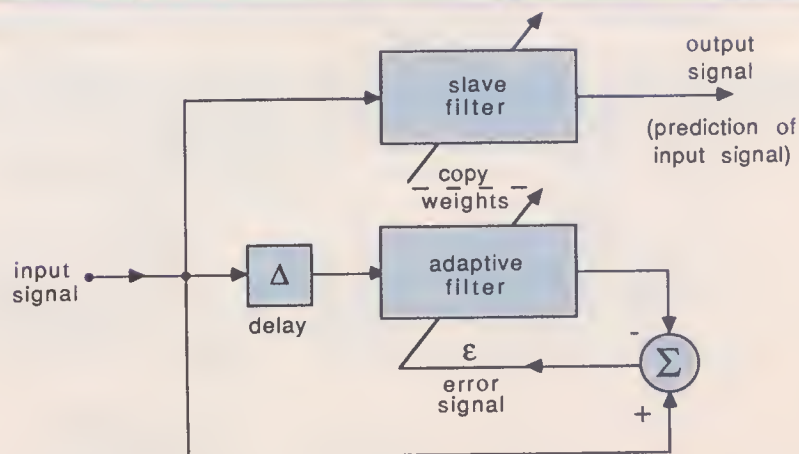


Figure 5. An adaptive statistical predictor.

problem in signal processing. Figure 6a shows a classical approach to this problem using optimal Wiener or Kalman filtering.³ The purpose of the optimal filter is to pass the signal s without distortion while stopping the noise n_0 . In general, this cannot be done perfectly. Even with the best

filter, the signal is distorted, and some noise goes through to the output.

Figure 6b shows another approach to the problem using adaptive filtering. This approach is viable only when an additional "reference input" is available containing noise n_1 , which is correlated with the

original corrupting noise n_0 . In Figure 6b, the adaptive filter receives the reference noise, filters it, and subtracts the result from the noisy "primary input," $s + n_0$. For this adaptive filter, the noisy input $s + n_0$ acts as the desired response. The "system output" acts as the error for the adaptive filter. Adaptive noise canceling generally performs much better than the classical approach since the noise is subtracted out rather than filtered out.

One might think that some prior knowledge of the signal s or of the noises n_0 and n_1 would be necessary before the filter could adapt to produce the noise-canceling signal y . A simple argument will show, however, that little or no prior knowledge of s , n_0 , or n_1 or of their interrelationships is required.

Assume that s , n_0 , n_1 and y are statistically stationary and have zero means. Assume that s is uncorrelated with n_0 and n_1 and suppose that n_1 is correlated with n_0 . The output is

$$\epsilon = s + n_0 - y \quad (1)$$

Squaring, one obtains

$$\epsilon^2 = s^2 + (n_0 - y)^2 + 2s(n_0 - y) \quad (2)$$

Taking expectations of both sides of Equation 2, and realizing that s is uncorrelated with n_0 and with y , yields

$$\begin{aligned} E[\epsilon^2] &= E[s^2] + E[(n_0 - y)^2] \\ &\quad + 2E[s(n_0 - y)] \\ &= E[s^2] + E[(n_0 - y)^2] \end{aligned} \quad (3)$$

Adapting the filter to minimize $E[\epsilon^2]$ will not affect the signal power $E[s^2]$. Accordingly, the minimum output power is

$$E_{\min}[\epsilon^2] = E[s^2] + E_{\min}[(n_0 - y)^2] \quad (4)$$

When the filter is adjusted so that $E[\epsilon^2]$ is minimized, $E[(n_0 - y)^2]$ is therefore also minimized. The filter output y is then a best least-squares estimate of the primary noise n_0 . Moreover, when $E[(n_0 - y)^2]$ is minimized, $E[(\epsilon - s)^2]$ is also minimized, since, from Equation 1,

$$(\epsilon - s) = (n_0 - y) \quad (5)$$

Adjusting or adapting the filter to minimize the total output power is tantamount to causing the output ϵ to be a best least-squares estimate of the signal s for the given structure and adjustability of the adaptive filter and for the given reference input.

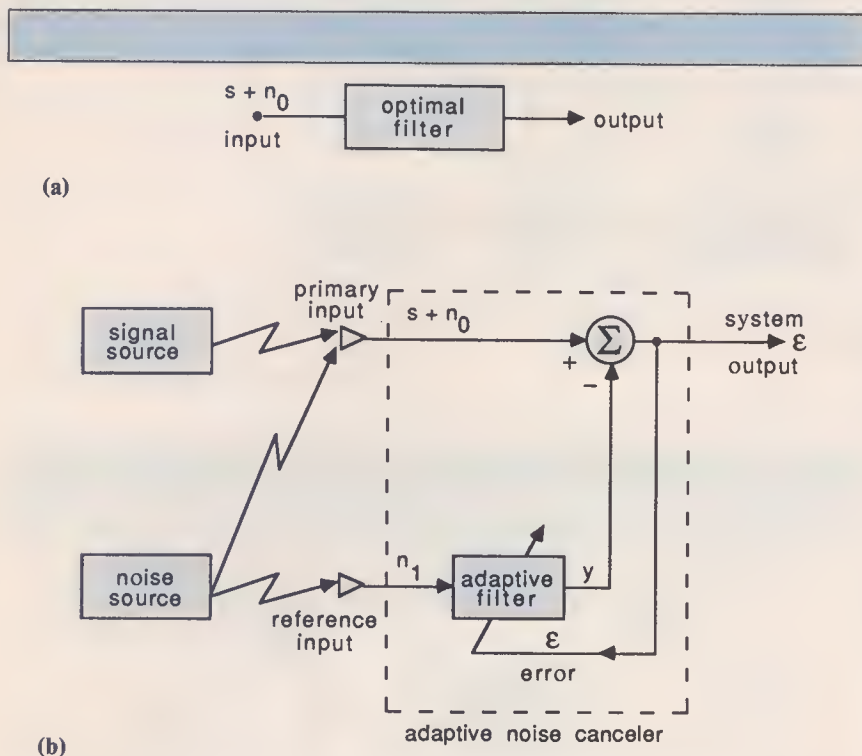


Figure 6. Separation of signal and noise: (a) classical approach; (b) adaptive noise-canceling approach.

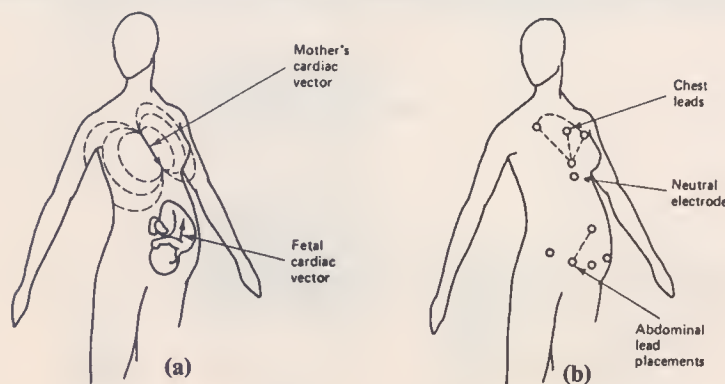


Figure 7. Canceling maternal heartbeat in fetal electrocardiography: (a) cardiac electric field vectors of mother and fetus; (b) placement of leads.

There are many practical applications for adaptive noise canceling techniques. One involves canceling interference from the mother's heart when attempting to record clear fetal electrocardiograms. Figure 7 shows the location of the fetal and maternal hearts and the placement of the input leads. The abdominal leads provide the primary input (containing fetal ECG and interfering maternal ECG signals), and the chest leads provide the reference input (containing pure interference, the maternal ECG). Figure 8 shows the results. The maternal ECG from the chest leads was adaptively filtered and subtracted from the abdominal signal, leaving the fetal ECG. This was an interesting problem since the fetal and maternal ECG signals had spectral overlap. The two hearts were electrically isolated and worked independently, but the second harmonic frequency of the maternal ECG was close to the fundamental of the fetal ECG. Ordinary filtering techniques would have great difficulty with this problem.

Example 4—adaptive echo canceling. Echo is a natural phenomenon in long-distance telephone circuits because of amplification in both directions and series coupling of telephone transmitters and receivers at each end of the circuit. Echo suppressors, used to break the feedback, give one-way communication to the party speaking first. To avoid switching effects and to permit simultaneous two-way transmission of voice and data, adaptive echo cancelers are replacing echo suppressors worldwide (see Figure 9).

In Figure 9, the delay boxes represent transmission delays in the long line. Note that separate circuits are normally used in each direction because the repeater amplifiers used to overcome transmission loss are one-way devices. Hybrid transformers

prevent incoming signals from coupling through the telephone set and passing as outgoing signals. Hybrids are balanced to do this by designing them for the average, local telephone circuit. Since each local circuit has its own length and electrical characteristics, the hybrid cannot do its job perfectly. Using an adaptive filter to cancel any incoming signal that might leak through the hybrid eliminates the possibility of echo. The circuit of Figure 9 works

well, allowing simultaneous two-way communication without echo.

Example 5—inverse modeling. Figure 4 showed use of an adaptive filter for direct modeling of an unknown system to obtain a close approximation to its impulse and frequency responses. By changing the configuration, it is possible to use the adaptive filter for inverse modeling to obtain the reciprocal of the unknown system's trans-

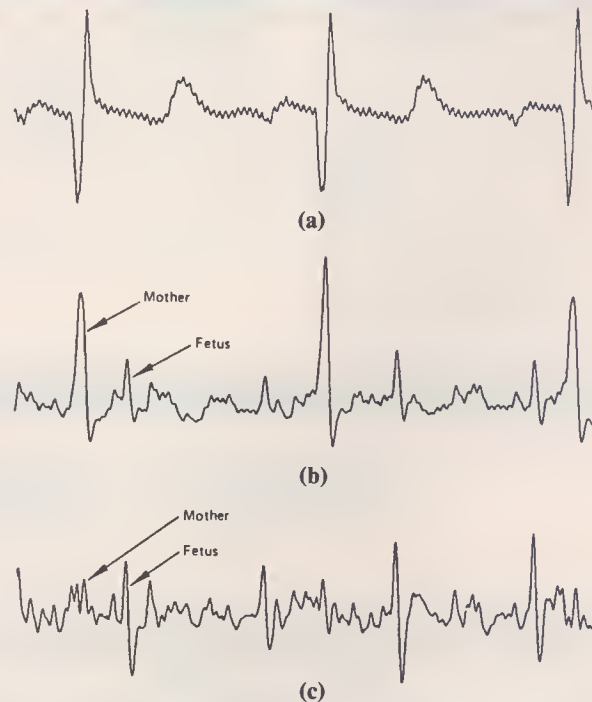


Figure 8. Result of fetal ECG experiment (bandwidth, 3-35 Hz; sampling rate, 256 Hz): (a) reference input (chest lead); (b) primary input (abdominal lead); (c) noise canceler output.

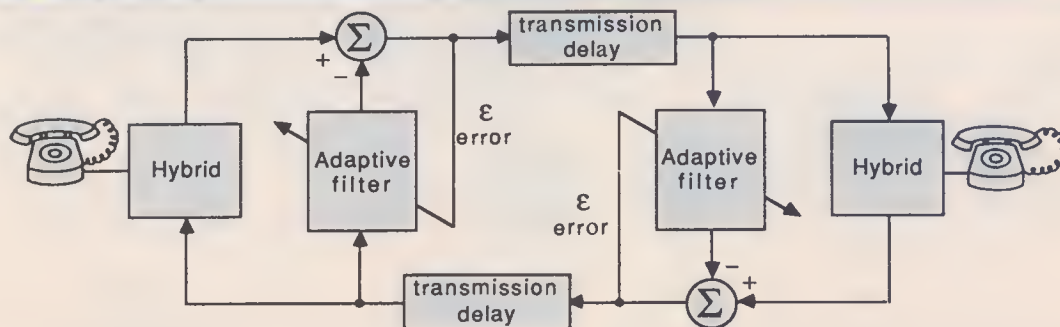


Figure 9. Long-distance system with adaptive echo cancellation.

fer function. The idea is illustrated in Figure 10. The output of the unknown system is the input to the adaptive filter. The unknown system's input delayed by Δ time units is the desired response of the adaptive filter. For simplicity, assume

that Δ is set to zero delay. To make the error small, the cascade of the unknown system and the adaptive filter needs a unity transfer function. Therefore, using an adaptive algorithm to make the error small causes the adaptive filter to develop a

transfer function that is the inverse of that of the unknown system.

If the response of the unknown system contains a delay or is nonminimum phase, allowing a nonzero delay Δ will be highly advantageous. Including Δ delays the inverse impulse response but yields a much lower mean-square error. In some applications, one would like to make this delay as small as possible. In other applications, this delay is of no concern except that it should be chosen to minimize the mean-square error. Applications for inverse modeling exist in the field of adaptive control, in geophysical signal processing, where it is called "deconvolution," and in telecommunications for channel equalization.

Example 6—channel equalization. Telephone channels, radio channels, and even fiber optic channels can have non-flat frequency responses and nonlinear phase responses in the signal passband. Sending digital data at high speed through these channels often results in a phenomenon called "intersymbol interference," caused by signal pulse smearing in the dispersive

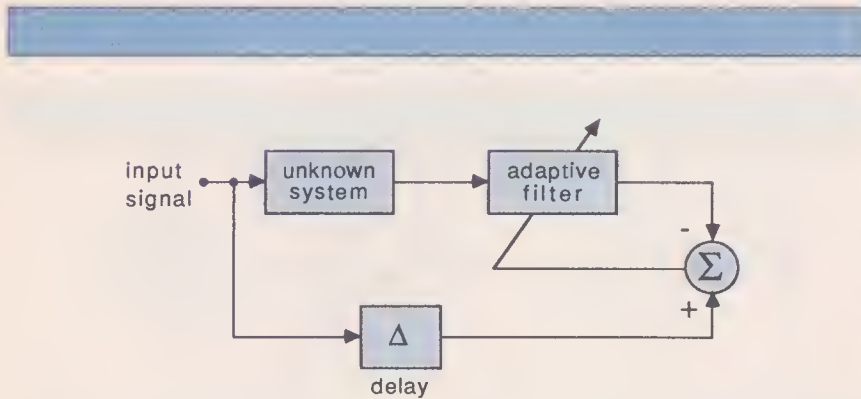


Figure 10. Inverse modeling.

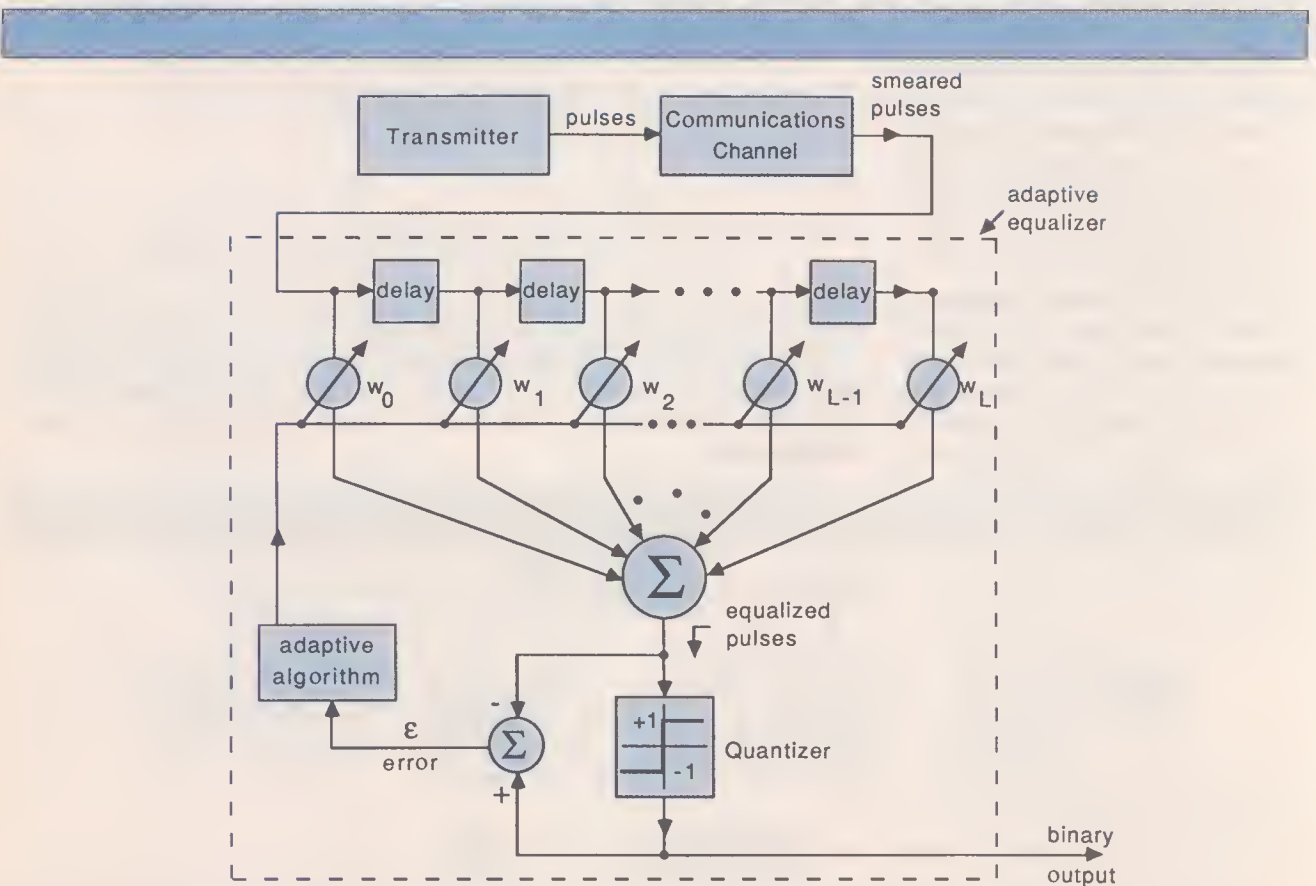


Figure 11. Adaptive channel equalizer with decision-directed learning.

medium. Equalization in data modems combats this phenomenon by filtering incoming signals. A modem's adaptive filter, by adapting itself to become a channel inverse, can compensate for the irregularities in channel magnitude and phase response.

The adaptive equalizer in Figure 11 consists of a tapped delay line with an adaptive linear combiner connected to the taps. Deconvolved signal pulses appear at the weighted sum, which is quantized to provide a binary output corresponding to the original binary data transmitted through the channel. The ALC and its quantizer comprise a single Adaline. Any least-squares algorithm can adapt the weights, but the telecommunications industry uses the LMS algorithm almost exclusively.

In operation, the weight at a central tap is generally fixed at unit value. Initially, all other weights are set to zero so that the equalizer has a flat frequency response and a linear phase response. Without equalization, telephone channels can provide quantized binary outputs that reproduce the transmitted data stream with error rates of 10^{-1} or less. As such, the quantized binary output can be used as the desired response to train the neuron. It is a noisy desired response initially. Sporadic errors cause adaptation in the wrong direction, but on average, adaptation proceeds correctly. As the neuron learns, noise in the desired response diminishes. Once the adaptive equalizer converges, the error

rate will typically be 10^{-6} or less. The method, called "decision-directed" learning, was invented by Robert W. Lucky of AT&T Bell Labs.⁴

Figure 12a shows the analog response of a telephone channel carrying high-speed binary pulse data. Figure 12b shows an "eye" pattern, which is the same signal after going through a converged adaptive equalizer. Equalization opens the eye and allows clear separation of +1 and -1 pulses. Using a modem with an adaptive equalizer enables transmitting about four times as much data through the same channel with the same reliability as without equalization.

Integrated services digital network (ISDN), a new concept now in development and deployment, makes high-speed digital communication possible through ordinary local copper telephone circuits. ISDN requires both adaptive equalization and adaptive echo canceling at each line termination. The number of adaptive filters to be used in the world's telecommunications plant will be massive.

Adaptive pattern recognition

The adaptive threshold element of Figure 3 can be used for pattern recognition and as a trainable logic device. It can be trained to classify input patterns into

two categories. For these applications, the zeroth weight, w_0 , has a constant input $x_0 = +1$ which does not change from input pattern to pattern. Varying the zeroth weight varies the threshold level of the quantizer.

Linear separability. With n binary inputs and one binary output, a single neuron of the type shown in Figure 3 is capable of implementing certain logic functions. There are 2^n possible input patterns. A general logic implementation would be capable of classifying each pattern as either +1 or -1, in accord with the desired response. Thus, there are 2^{2^n} possible logic functions connecting n inputs to a single output. A single neuron is capable of realizing only a small subset of these functions, known as the linearly separable logic functions.⁵ These are the set of logic functions that can be obtained with all possible settings of the weight values.

Figure 13 shows a two-input neuron, and Figure 14 shows all of its possible binary inputs in pattern vector space. In this space, the coordinate axes are the components of the input pattern vector. The neuron separates the input patterns into two categories, depending on the values of the input-signal weights and the bias weight. A critical thresholding condition occurs when the analog response y equals zero:

$$y = x_1w_1 + x_2w_2 + w_0 = 0 \quad (6)$$

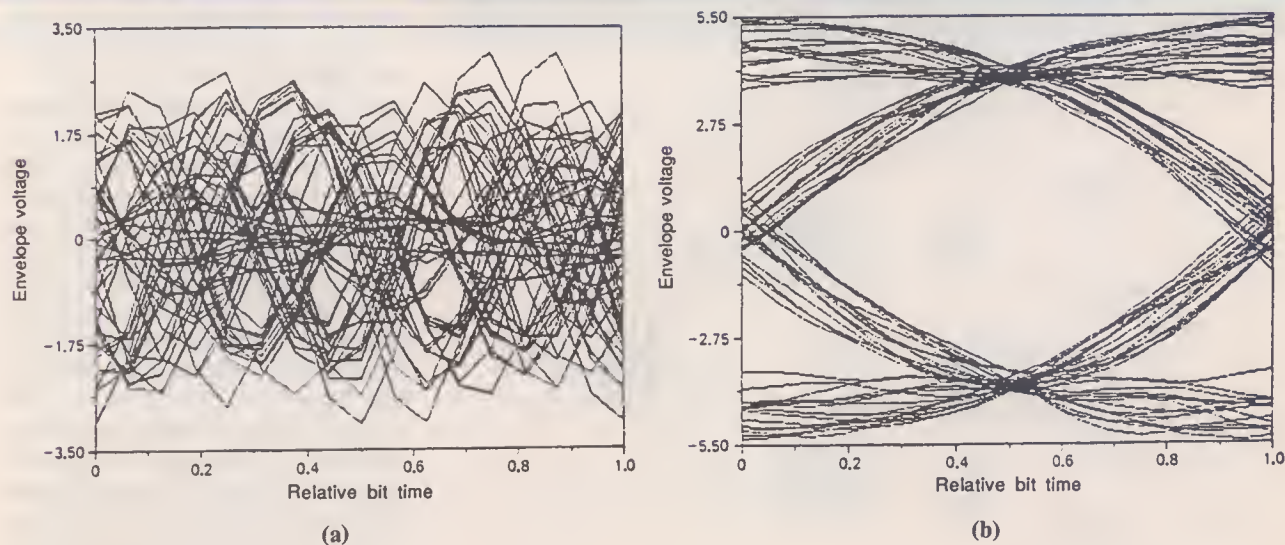


Figure 12. Eye patterns produced by overlaying cycles of the received waveform: (a) before equalization; (b) after equalization.

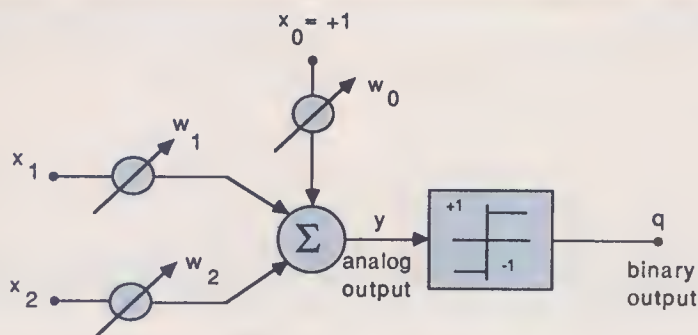


Figure 13. A two-input neuron.

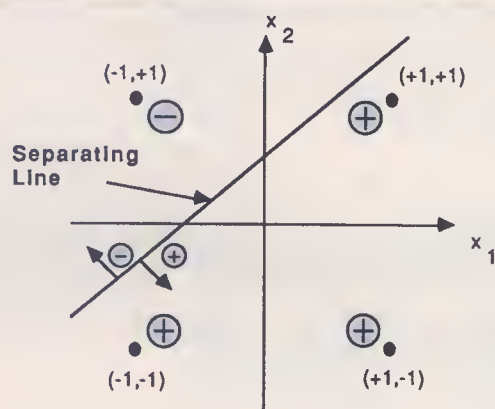


Figure 14. Separating line in pattern space.

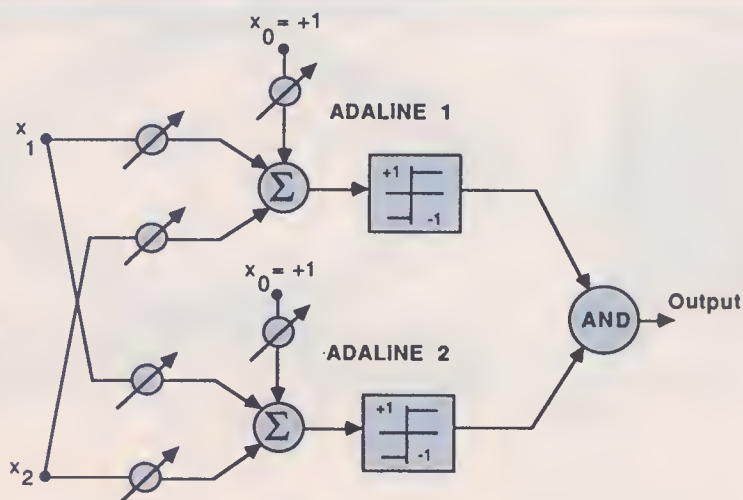


Figure 15. A two-neuron form of Madaline (many Adalines).

$$\therefore x_2 = -(w_0/w_2) - (w_1/w_2) x_1 \quad (7)$$

Figure 14 graphs this linear relation, which comprises a separating line having slope and intercept of

$$\text{slope} = -(w_1/w_2) \quad (8)$$

$$\text{intercept} = -(w_0/w_2)$$

The three weights determine slope, intercept, and the side of the separating line that corresponds to a positive output. The opposite side of the separating line corresponds to a negative output.

The input/output mapping obtained in Figure 14 illustrates an example of a linearly separable function. An example of a nonlinearly separable function with two inputs is the following:

$$\begin{aligned} (+1, +1) &\rightarrow +1 \\ (+1, -1) &\rightarrow -1 \\ (-1, -1) &\rightarrow +1 \\ (-1, +1) &\rightarrow -1 \end{aligned} \quad (9)$$

No single line exists that can achieve this separation of the input patterns.

With two inputs, a single neuron can realize almost all possible logic functions. With many inputs, however, only a small fraction of all possible logic functions are linearly separable. The single neuron can realize only linearly separable functions and generally cannot realize most functions. However, combinations of neurons or networks of neurons can be used to realize nonlinearly separable functions.

Nonlinear separability—Madaline networks. In the early 1960s at Stanford, Ridgway⁶ initiated an approach to the implementation of nonlinearly separable logic functions. He connected retinal inputs to adaptive neurons in a single layer and, in turn, connected their outputs to a fixed logic device providing the system output. Methods for adapting such nets were developed at that time. In the example network shown in Figure 15, two Adalines are connected to an AND logic device to provide an output. Systems of this type were called Madalines (many Adalines). Today such systems would be called small neural nets.

With weights suitably chosen, the separating boundary in pattern space for the system of Figure 15 would be as shown in Figure 16. This separating boundary implements the nonlinearly separable logic function of Equation 9.

Madalines were constructed with many more inputs, with many more neurons in

the first layer, and with fixed logic devices such as AND, OR, and MAJority voter in the second layer. Those three functions, as illustrated in Figure 17, are in themselves threshold logic functions. The weights given will implement these functions, but the weight choices are not unique.

Layered neural nets. The Madalines of the 1960s had adaptive first layers and fixed threshold functions for the second (output) layers.^{6,7} The feed-forward neural nets of the 1980s have many layers, and all layers are adaptive. The back-propagation method of Rumelhart et al.⁸ is perhaps the best-known example of a multilayer network. A three-layer feed-forward adaptive network is illustrated in Figure 18.

Adapting the neurons in the output layer is simple, since the desired responses for the entire network (given with each input training pattern) are the desired responses for the corresponding output neurons. Given the desired responses, adaptation of the output layer can be a straightforward exercise of the LMS algorithm. The fundamental difficulty associated with adapting a layered network lies in obtaining desired responses for the neurons in the layers other than the output layer. The back-propagation algorithm (first reported by Werbos⁹ and later rediscovered by Parker¹⁰ and by Rumelhart et al.⁸) is one method for establishing desired responses for the neurons in the "hidden layers," those layers whose neuronal outputs do not appear directly at the system output (see Figure 18). There is nothing unique about the choice of desired outputs for the hidden layers.

Generalization in layered networks is a key issue. The question is, how well do multilayered networks perform with inputs for which they were not specifically trained? The question of generalization is important, and theorists are developing some good examples where useful generalizations take place. Many different algorithms may be needed for the adaptation of multilayered networks to produce required generalizations. Without generalization, neural nets will be of little engineering significance. Merely learning the training patterns can be accomplished by storing these patterns and their associated desired responses in a look-up table.

The layered networks of Rumelhart et al.⁸ use neuronal elements like the Adaline of Figure 3, except that the quantizer

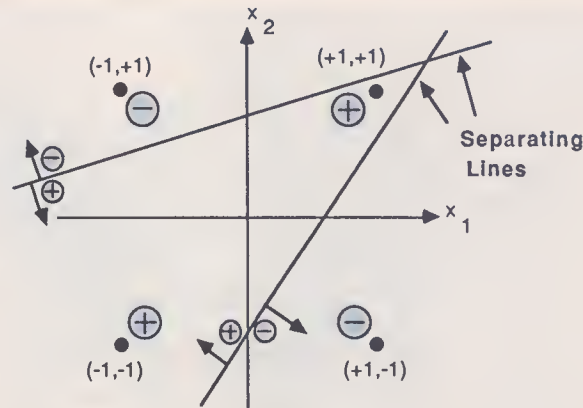


Figure 16. Separating boundaries for the Madaline of Figure 15.

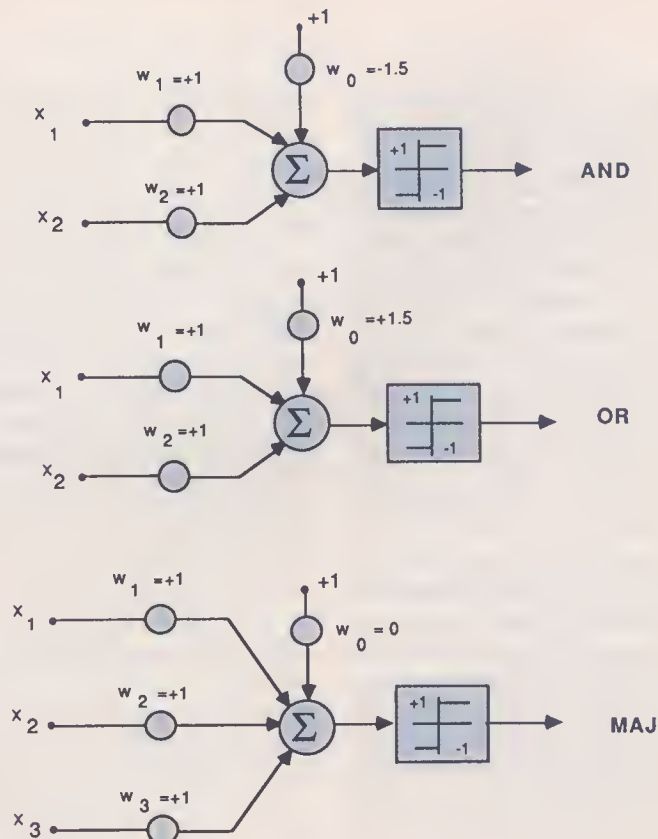


Figure 17. Neuronal implementation of AND, OR, and MAJ logic functions.

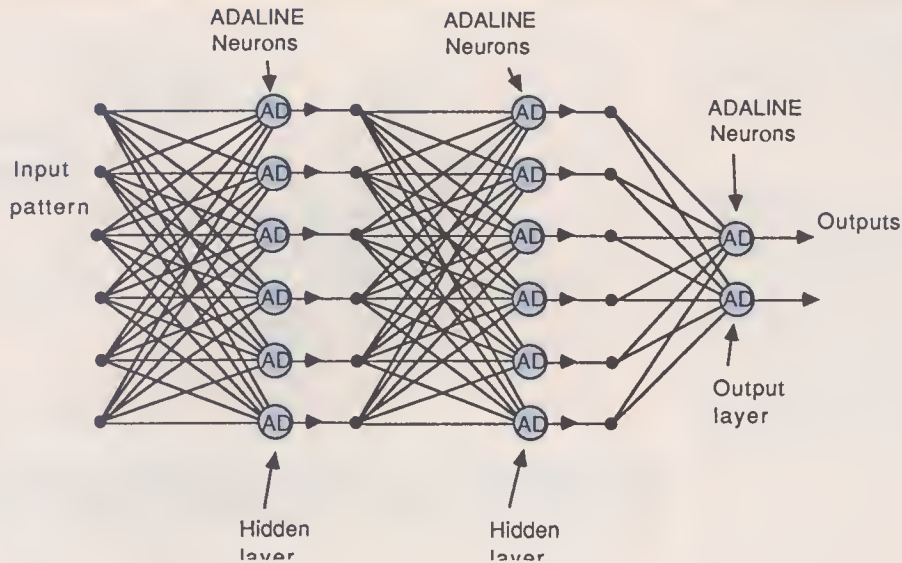


Figure 18. A three-layer adaptive neural network.

or threshold device is a soft-limiting "sigmoid" function rather than the hard-limiting "signum" function of the Adaline. The various back-propagation algorithms for adapting layered networks of neurons require differentiability along the network's signal paths and cannot work with the Adaline element's hard limiter. The sigmoid function has the necessary differentiability. However, it presents implementational difficulties if the neural net is ultimately constructed digitally. For this reason, we developed a new algorithm for adapting layered networks of Adaline neurons with hard-limiting quantizers. The new algorithm is an extension of the original Madaline adaptation rule^{6,7} and is called Madaline rule II or MR II. The idea is to adapt the network to properly respond to the newest input pattern while minimally disturbing the responses already trained-in for the previous input patterns. Unless this principle is practiced, it is difficult for the network to simultaneously store all of the required pattern responses.

LMS or Widrow-Hoff delta rule for the single neuron. The LMS algorithm applied to the adaptation of the weights of a single neuron embodies a *minimal distur-*

bance principle. This algorithm can be written as

$$W_{k+1} = W_k + \alpha \epsilon_k X_k / |X_k|^2 \quad (10)$$

The time index or adaption cycle number is k . W_{k+1} is the next value of the weight vector, W_k is the present value of the weight vector, X_k is the present input pattern vector, and ϵ_k is the present error (that is, the difference between the desired response d_k and the analog output before adaptation). Applying the above recursion formula to each adaption cycle reduces the error by the fraction α . That is, at the k th cycle, the error is

$$\epsilon_k = d_k - X_k^T W_k \quad (11)$$

Changing the weights changes (reduces) the error:

$$\begin{aligned} \Delta \epsilon_k &= \Delta(d_k - X_k^T W_k) \\ &= -X_k^T \Delta W_k \end{aligned} \quad (12)$$

In accordance with the LMS rule (Equation 10), the weight change is as follows:

$$\begin{aligned} \Delta W_k &= W_{k+1} - W_k \\ &= \alpha \epsilon_k X_k / |X_k|^2 \end{aligned} \quad (13)$$

Combining Equations 12 and 13, we obtain

$$\begin{aligned} \Delta \epsilon_k &= -X_k^T \alpha \epsilon_k X_k / |X_k|^2 \\ &= -\alpha \epsilon_k X_k^T X_k / |X_k|^2 \\ &= -\alpha \epsilon_k \end{aligned} \quad (14)$$

Therefore, the error is reduced by a factor of α as the weights are changed while holding the input pattern fixed. Putting in a new input pattern starts the next adaptation cycle. The next error is then reduced by a factor of α , and the process continues.

The choice of α controls stability and speed of convergence.¹ Stability requires that

$$2 > \alpha > 0 \quad (15)$$

Making α greater than 1 generally does not make sense, since the error would be over-corrected. Total error correction comes with $\alpha = 1$. A practical range for α is

$$1.0 > \alpha > 0.1 \quad (16)$$

The weights change proportionately with their inputs in accordance with the LMS algorithm (see Equation 10). With the usual binary inputs 1 and 0, no adaptation occurs for weights with 0 inputs.

Thus, the symmetric inputs $+1$ and -1 are generally preferred.

To verify that the LMS rule embodies the minimal disturbance principle, refer to Equation 13. The weight-change vector ΔW_k is chosen to be parallel to the input pattern vector X_k . From Equation 12, the change in the error is equal to the negative dot product of X_k with ΔW_k , thus achieving the needed error correction with the smallest magnitude of weight vector change. When adapting to respond properly to a new input pattern, the responses to previous training patterns are therefore minimally disturbed, on the average. The algorithm also minimizes mean square error,¹ the property for which it is best known.

Adaptation of layered neural nets by the MRIL rule. The minimal disturbance principle can be applied to the adaptation of Figure 18's layered neural network. Presenting an input pattern and its associated desired responses to the network, the training objective is to reduce the number of errors to as low a level as possible. Accordingly, when the first training pattern is presented, the first layer will adapt as required to reduce the number of response errors at the final output layer. In accordance with the minimal disturbance principle, the first-layer neuron whose analog response is closest to zero is given a trial adaptation in the direction to reverse its binary output. When the reversal takes place, the second-layer inputs change, the second-layer outputs change, and consequently the network outputs change. A check is made to see if this reduces the number of output errors for the current input pattern. If so, the trial change is accepted. If not, the weights are restored to their previous values, and the first-layer neuron whose analog response is next closest to zero is trial adapted, reversing its response. If this reduces the number of output errors, the change is accepted. If not, the weights are restored, and one goes on to adaptively switch the neuron with an analog response next closest to zero, and so on, disturbing the neurons as little as possible. After adapting all neurons whose output reversals reduced the number of output errors, neurons are then chosen in pair combinations and trial adaptations are made which can be accepted if output errors are reduced. After adapting the first-layer neurons in singles, pairs, triples, etc., up to a predetermined limit in combination size (simulation results indicate pairwise trials are

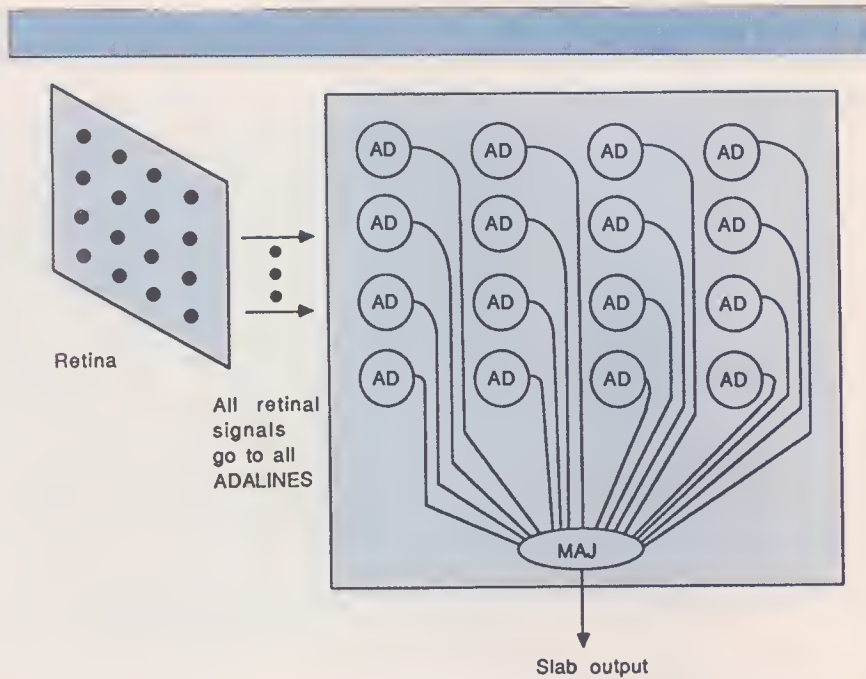


Figure 19. One slab of a left-right, up-down translation invariant network.

sufficient in layers having up to 25 Adalines), the second layer is adapted to further reduce the number of network output errors. The method of choosing the neurons to be adapted in the second layer is the same as that for the first layer. If further error reduction is needed, the output layer is then adapted. This is straightforward, since the output-layer desired responses are the given desired responses for the entire network. After adapting the output layer, the responses will be correct. The next input pattern vector and its associated desired responses are then applied to the neural network, and the adaptive process resumes.

When training the network to respond correctly to the various input patterns, the "golden rule" is *give the responsibility to the neuron or neurons that can most easily assume it*. In other words, *don't rock the boat* any more than necessary to achieve the desired training objective. (Simulation results using this minimal-disturbance MRIL algorithm are presented later.)

Application of layered networks to pattern recognition. It would be useful to devise a neural net configuration that could be trained to classify an important

set of training patterns as required, but have these network responses be invariant to translation, rotation, and scale change of the input pattern within the field of view. It should not be necessary to train the system with the specific training patterns of interest in all combinations of translation, rotation, and scale.

The first step is to show that a neural network having these properties exists. (The invariance methods that follow are extensions of results reported earlier by Widrow.²) The next step is to obtain training algorithms to achieve the desired objectives.

Invariance to up-down, left-right pattern translation. Figure 19 shows a planar network configuration (a "slab" of neurons) that could be used to map a retinal image into a single-bit output so that, with proper weights in the network's neurons, the response will be insensitive to left-right and/or up-down translation. The same slab structure can be replicated, with different weights, to allow the retinal pattern to be independently mapped into additional single-bit outputs, all insensitive to left-right, up-down translation.

Figure 20 illustrates the general idea. A retinal image having a given number of

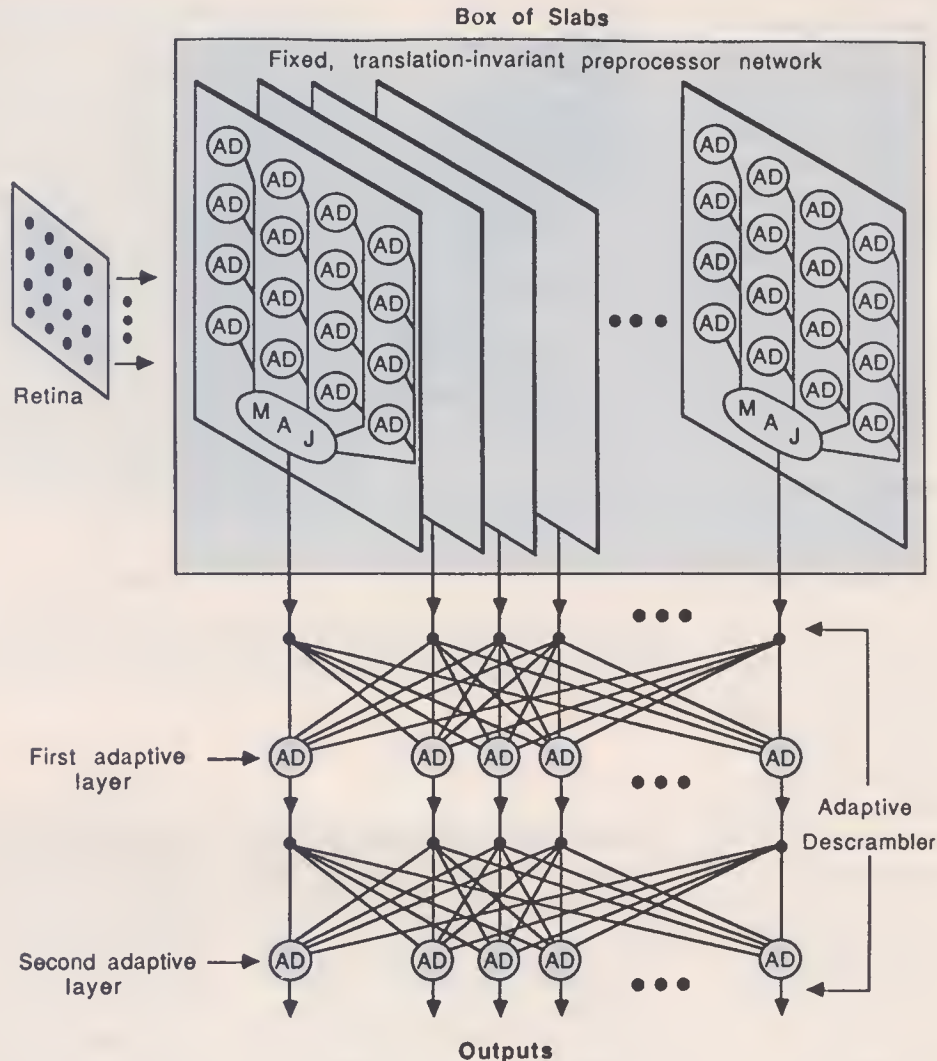


Figure 20. A translation-invariant preprocessor network and an adaptive two-layer descrambler network.

pixels can be mapped through an array of slabs into a different image having the same, more, or fewer pixels, depending on the number of slabs used. In any event, the mapped image is insensitive to up-down, left-right translation of the original image. The mapped image in Figure 20 is fed to a set of Adaline neurons that can be easily trained to provide output responses to the original image as required. This amounts to a "descrambling" of the preprocessor's outputs. The descrambler's output responses classify the original input images and, at the same time, are insensitive to their left-right, up-down translations.

In the systems of Figures 19 and 20, the elements labeled "AD" are Adalines.

Those labeled "MAJ" are majority vote-takers. (If the number of input lines to MAJ is even and there is a tie vote, these elements are biased to give a positive response.) The AD elements are adaptive neurons and the MAJ elements are fixed neurons, as in Figure 17.

In the system shown in Figure 19, the structuring of the weights so that the output is insensitive to left-right and up-down translation needs further explanation. Let the weights of each Adaline be arranged in a square array and the corresponding retinal pixels arrayed in a square pattern. Let the square matrix (W_1) designate the array of weights of the upper-left Adaline,

and let $T_{D1}(W_1)$ be the array of weights of the next lower Adaline. The operator T_{D1} represents "translate down one," so the second set of weights is the same as the top-most set, but translated down en masse by one pixel. The bottom row wraps around to comprise the top row. The patterns on the retina itself wrap around on a cylinder when they undergo translation. The weights of the next lower Adaline are $T_{D2}(W_1)$, and those of the next lower Adaline are $T_{D3}(W_1)$. Returning to the upper-left Adaline, let its neighbor to the right be designated by $T_{R1}(W_1)$, with T_{R1} being a "translate right one" operator. The pattern of weights for the entire array of Adalines in Figure 19 is

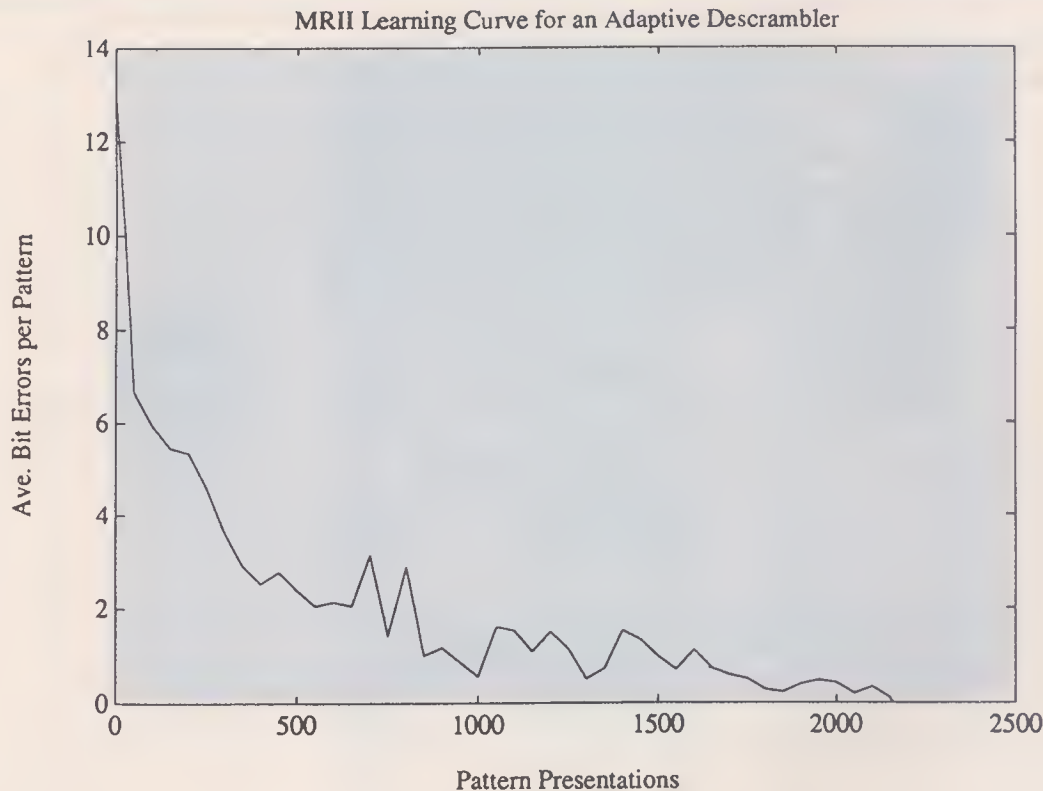


Figure 21. Learning curve for a two-layer, 25×25 adaptive descrambler.

$$\begin{array}{cccc}
 (W_1) & T_{R1}(W_1) & T_{R2}(W_1) & T_{R3}(W_1) \\
 T_{D1}(W_1) & T_{R1}T_{D1}(W_1) & T_{R2}T_{D1}(W_1) & T_{R3}T_{D1}(W_1) \\
 T_{D2}(W_1) & T_{R1}T_{D2}(W_1) & T_{R2}T_{D2}(W_1) & T_{R3}T_{D2}(W_1) \\
 T_{D3}(W_1) & T_{R1}T_{D3}(W_1) & T_{R2}T_{D3}(W_1) & T_{R3}T_{D3}(W_1)
 \end{array}$$

(17)

As the input pattern moves up, down, left, or right on the retina, the roles of the various Adalines interchange. Since all Adaline outputs are equally weighted by the MAJ element, translating the input pattern up-down and/or left-right on the retina has no effect on the MAJ element output.

The set of "key" weights (W_1) can be randomly chosen. Once chosen, they can be translated according to Equation 17 to fill out the array of weights for the system of Figure 19. This array of weights can be incorporated as the weights for the first slab of Adalines shown in Figure 20. The weights for the second slab would require the same translational symmetries, but be based on a different randomly chosen set of key weights (W_2). The mapping func-

tion of the second slab would therefore be distinct from that of the first slab.

The translational symmetries in the weights called for in Figure 19 could be fixed and manufactured in, or they could be arrived at through training. If, when designing an application-specific pattern recognition system, one knew that translational invariance would be required, it would make sense to manufacture the appropriate symmetry into a fixed weight system, leaving only the final-output Adaline layers plastic and trainable (see Figure 20). Such a preprocessor would definitely work, would provide very high speed response without scanning and searching for pattern location and alignment, and would be an excellent application of neural nets.

Invariance to rotation. Figure 20 represents a system for preprocessing retinal patterns with a translation-invariant fixed neural net followed by a two-layer adaptive descrambler net. The system can

be expanded to incorporate rotational invariance. Suppose that all input patterns can be presented in "normal" vertical orientation, approximately centered within the field of view of the retina. Suppose further that all input patterns can be presented when rotated from normal by 90, 180, and 270 degrees. Thus, each pattern can be presented in all four rotations and in all possible left-right, up-down translations. The number of combinations would be large. The problem is to design a neural net preprocessor that is invariant to translation and to rotation by 90 degrees.

Begin with a single slab of Adaline elements, as shown in Figure 19, producing a majority output that is insensitive to translation of the input pattern on the retina. Next, replicate this slab four-fold, and let the majority outputs feed into a single majority output element. In the first slab, (W_1) designates the upper-left Adaline's matrix of weights. (See Equation 17 for the weight matrices of all first-slab Adalines.)

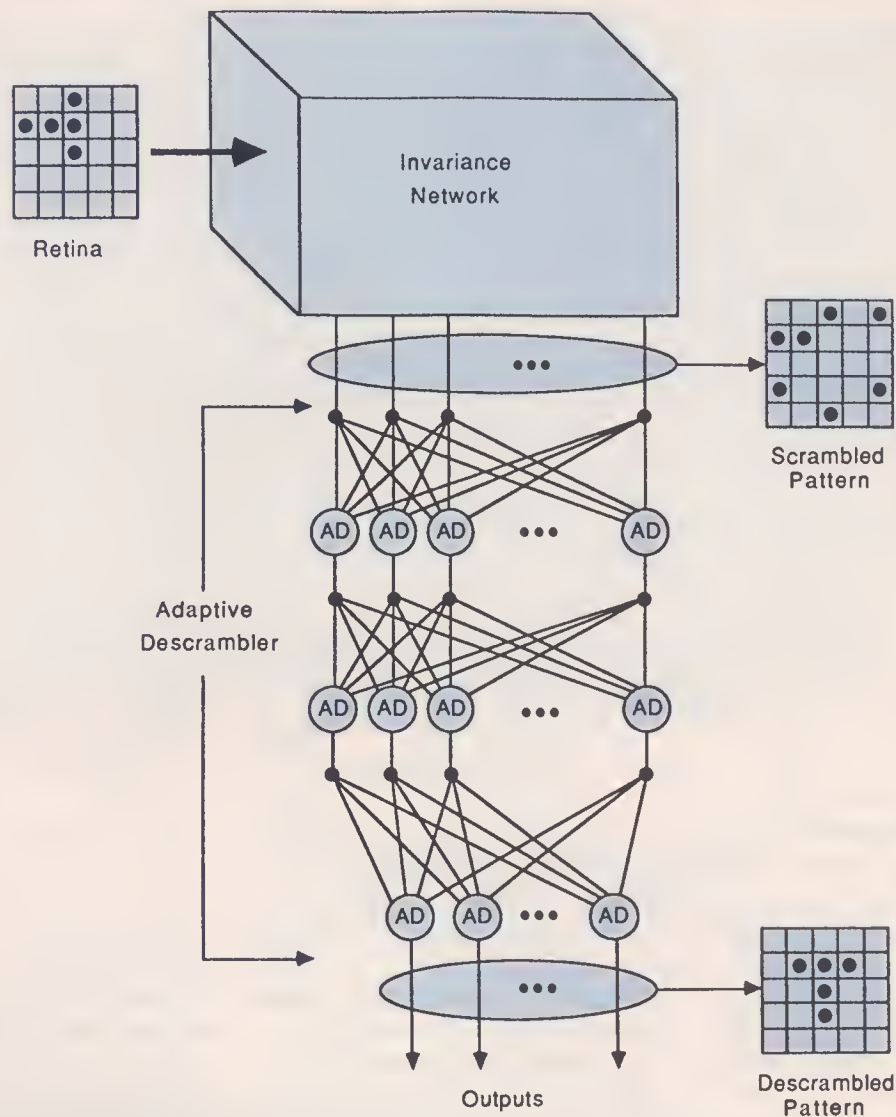


Figure 22. A Madaline system for pattern recognition.

In the second slab, the upper-left Adaline's weight matrix corresponds to the first-slab weight matrix rotated 90 degrees clockwise. This can be designated by $R_{C1}(W_1)$, and the corresponding third- and fourth-slab weight matrices can be designated by $R_{C2}(W_1)$ and $R_{C3}(W_1)$. Thus, the weight matrices of the upper-left Adalines begin with (W_1) in the first slab, and are rotated clockwise by 90 degrees in the second slab, by 180 degrees in the third slab, and by 270 degrees in the fourth slab. The weight matrices of all slabs are translated right and down, in the fashion of Equation 17, starting with the Adalines in the upper left-

hand corner. For example, the array of weight matrices for the second slab is

$$\begin{array}{cccc}
 R_{C1}(W_1) & T_{R1}R_{C1}(W_1) & T_{R2}R_{C1}(W_1) & T_{R3}R_{C1}(W_1) \\
 T_{D1}R_{C1}(W_1) & T_{R1}T_{D1}R_{C1}(W_1) & T_{R2}T_{D1}R_{C1}(W_1) & T_{R3}T_{D1}R_{C1}(W_1) \\
 T_{D2}R_{C1}(W_1) & T_{R1}T_{D2}R_{C1}(W_1) & T_{R2}T_{D2}R_{C1}(W_1) & T_{R3}T_{D2}R_{C1}(W_1) \\
 T_{D3}R_{C1}(W_1) & T_{R1}T_{D3}R_{C1}(W_1) & T_{R2}T_{D3}R_{C1}(W_1) & T_{R3}T_{D3}R_{C1}(W_1)
 \end{array} \quad (18)$$

Clearly, translating the pattern on the retina does not change the majority output response. Rotating the pattern 90 degrees causes an interchange of the roles of the slabs in making their responses, but, since the output majority element weights them

equally, the output response is unchanged. Insensitivity to 45-degree rotation can be accomplished by using more slabs; thus, a complete neural network providing invariance to rotation and translation could be constructed. Each translation-invariant slab of Figure 20 would need to be replaced by the rotation-invariant multiple slab and majority-element system described above.

Invariance to scale. The same principles can be used to design invariance nets that are insensitive to scale or pattern size. By establishing a "point of expansion" on the retina so that input patterns can be

expanded or contracted with respect to this point, two Adalines can be trained to give similar responses to patterns of two different sizes if the weight matrix of one expands (or contracts) about the point of expansion like the patterns themselves. The amplitude of the weights must be scaled in inverse proportion to the square of the linear dimension of the retinal pattern. By adding many more slabs, the invariance net can be built around this idea to be insensitive to pattern size as well as to translation and rotation. (Implementation would, of course, require the abundance and low cost of VLSI electronics.)

Simulation results. The system of Figure 20 was computer simulated. The training set consisted of 36 patterns, each arranged on a 5×5 pixel retina in "standard" position. Twenty-five slabs, each with 25 Adalines having weights fixed according to Equation 17, were used in the translation-invariant preprocessor. The preprocessor output represented a scrambled version of the input pattern. The nature of this scrambling was determined by the choice of the key weight matrices (W_1), . . . , (W_{25}). These key weights were chosen randomly, the only requirement was that the input pattern to preprocessor output map be one-to-one. (This choice of weights produced a very noise-intolerant mapping. We are investigating methods of training in the key weights, using MRII to customize them to the training set.)

We used MRII to train the descrambler, a two-layer system with 25 Adalines in each layer. The initial descrambler weights were chosen randomly and independently, distributed uniformly on the interval $(-1, +1)$. Patterns were presented in random order, each pattern being equally likely of being the next presented. The desired response used was the training pattern in standard position. The system as a whole would then recognize any trained-in pattern in any translated position on the input retina and reproduce it in standard position at the output. Figure 21, a typical descrambler learning curve, graphs the number of incorrect pixels at the output, averaged over the training set, every 50 pattern presentations.

Much work on MRII remains to be done, including detailed studies of its convergence properties and its ability to produce generalizations. Preliminary results are very encouraging. Applying the algorithm to problems will lead to insights that will, we hope, allow a mathematical analysis of the algorithm.

The concept of using an invariance preprocessor followed by a descrambler is a potentially powerful one. We plan to apply the concept to a speech recognition problem. When a word is spoken by different people or even by the same person at different times, the sounds produced differ greatly but remain recognizable as the same word—at least to a human. Therefore, those sounds must have properties that are invariant from utterance to utterance. We believe a system similar to the one in Figure 20 would be useful in developing a multiuser speech recognition system.

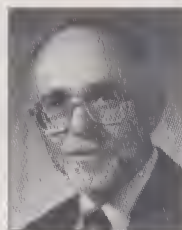
The general pattern-recognition concept we've described involves use of an invariance net followed by a trainable classifier. Figure 22 illustrates the key ideas. The invariance net can be trained or designed to produce a set of outputs that are insensitive to translation, rotation, scale change, etc., of the retinal pattern. These outputs are scrambled, but the adaptive layers can be trained to descramble them and reproduce the original patterns in "standard" position, orientation, and scale. Multilayer adaptation algorithms are essential to making such a scheme work, and we've devised a new Madaline adaptation rule—MRII—for that purpose. Our preliminary experimental results indicate that it works and is effective. □

References

1. B. Widrow and S.D. Stearns, *Adaptive Signal Processing*, Prentice Hall, Englewood Cliffs, N.J., 1985.
2. B. Widrow, "Generalization and Information Storage in Networks of Adaline 'Neurons'," in *Self-Organizing Systems 1962*, M.C. Yovitz, G.T. Jacobi, and G. Goldstein, eds., Spartan Books, Washington, DC, 1962, pp. 435-461.
3. T. Kailath, *Lectures on Wiener and Kalman Filtering*, Springer Verlag, New York, 1981.
4. R.W. Lucky, "Automatic Equalization for Digital Communication," *Bell Syst. Tech. J.*, Vol. 44, Apr. 1965, pp. 547-588.
5. P.M. Lewis II and C.L. Coates, *Threshold Logic*, John Wiley and Sons, New York, 1967.
6. W.C. Ridgway III, *An Adaptive Logic System with Generalizing Properties*, PhD thesis, Stanford Electronics Labs. Rep. 1556-1, Stanford University, Stanford, Calif., Apr. 1962.
7. N. Nilsson, *Learning Machines*, McGraw-Hill, New York, 1965.
8. D.E. Rumelhart and J.L. McClelland, *Parallel Distributed Processing*, Vol. I and II, MIT Press, Cambridge, Mass., 1986.

9. P. Werbos, *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences*, PhD thesis, Harvard, Aug. 1974.

10. D.B. Parker, *Learning Logic*, Tech. Rep. TR-47, Center for Computational Research in Economics and Management Science, MIT, Apr. 1985.



Bernard Widrow is a professor of electrical engineering at Stanford University. Before joining the Stanford faculty in 1959, he was with the Massachusetts Institute of Technology, Cambridge. He is presently engaged in research and teaching in systems theory, pattern recognition, adaptive filtering, and adaptive control systems. He is an associate editor of the journals *Adaptive Control and Signal Processing*, *Neural Networks*, *Information Sciences*, and *Pattern Recognition* and coauthor with S.D. Stearns of *Adaptive Signal Processing* (Prentice Hall).

Widrow received the SB, SM and ScD degrees from MIT in 1951, 1953, and 1956. He is a member of the American Association of University Professors, the Pattern Recognition Society, Sigma Xi, and Tau Beta Pi. A fellow of the IEEE and of the American Association for the Advancement of Science, Widrow received the IEEE Alexander Graham Bell Medal in 1986 for exceptional contributions to the advancement of telecommunications.



Rodney Winter is a captain in the United States Air Force and a graduate student attending Stanford University through the Civilian Institutes Program of the Air Force Institute of Technology. His research interests include signal processing, pattern recognition, and adaptive systems. His doctoral thesis topic is the application of neural networks to optimal nonlinear filtering. His prior duties were those of a pilot, most recently in the F-106 interceptor.

Winter received his BSEE and MSEE degrees from Purdue University in 1977. He is a member of Eta Kappa Nu and a student member of the IEEE.

The authors' address is Information Systems Laboratory, Dept. of Electrical Engineering, Stanford University, Stanford, CA 94305.

MITRE



System Engineering

Radar Systems

Digital Signal Processing • Radar System Surveillance Techniques • Radar System Analyses • Anti Jam Analyses • Intelligence Operations Analyses • Embedded Computer Hardware and Software

System Software

On VM • CMS • VCNA • RSCS • VS1 • RSTS • RSX 11-M • UNIX® • VAX/VMS • Design, Implement and Document System Software • Performance Monitoring • Software and Hardware Evaluations • Enhance User Facilities • Applications Software—Corporate, Financial & Administrative Applications; MVS/COMPLETE/ADABAS Environment; PL/1 & NATURAL Programming Languages

Software Technology

Knowledge Based Expert Systems • Automatic Programming • Fault Tolerant Systems • Reusable Software

Software Engineering

Prototype Development • Performance Simulation • Ada® Compiler Evaluation • Software Cost Estimation • Project Management Tools • Artificial Intelligence

Civil Programs—McLean, VA Only

Computer Systems Architecture & Engineering • Systems Acquisition Management • Systems Planning & Analysis • Requirements Analysis & Definition • Computer Technology • Real-Time Computer Software Analysis & Sizing • Display Technology • Bus-Oriented System Architectures • Local Network Design • Radar System Design

Communications

System Design and Analyses • Digital Communications • Microprocessor Applications • Communications Processors (Hardware and Software) • Local Area Network • Protocol Development/Evaluation • Modulation/Coding Techniques • Secure Communications • Systems Test and Evaluation • Switching and Control Systems • Satellite and Terrestrial Transmission • Survivability/Vulnerability and Electromagnetic Pulse • Line of Sight, Tropo, Millimeter Wave and Fiber-Optics

Command And Control

Air Defense Systems (Deployable and Fixed) • Tactical Air Traffic Control Systems (Deployable and Fixed) • Foreign Air Command and Control • Missile Warning Systems • Satellite Surveillance Systems • Systems Analyses and Specifications • Software Development

Computer Systems

Requirements Analyses • Systems Analyses • Technology Assessment • System Acquisition • Performance Analyses • Simulation and Analytical Modeling • Artificial Intelligence • Image Processing • Fault Tolerant Systems • Ada • Software Cost Estimation • Computer Security • Software Metrics • Distributed Data Base Systems • Program Verification

Systems Architecture

Advanced Systems Design • Advanced Planning • Intersystems Engineering • Functional/Operational Analyses • Systems Inter-Operability • Cost Analyses

UNIX is registered trademark of AT&T Bell Laboratories
Ada is a registered trademark of the U.S. DoD

U.S. CITIZENSHIP REQUIRED.

C³I Careers with Global Significance

As a System Engineer at The MITRE Corporation, the significance of your engineering career will increase. The immense importance of our work, providing engineering expertise for America's and The Free World's Command, Control, Communications and Intelligence (C³I) systems, offers you truly vital projects having significant impact on the large scale electronics systems of the future.

Artificial intelligence, fiber optics, over-the-horizon radar, data encryption and security, local area networks, and civil programs involving air transportation and advanced information systems are just some of the 150 projects underway at MITRE's suburban Washington, D.C. and Boston locations.

The significance of MITRE's work will become quickly apparent when you discover the finest professional environment of your career, the close contact you'll have with world-leading experts, and the unparalleled resources to support your best work. You'll also enjoy outstanding opportunity for project

mobility, advancement through MITRE's dual career ladder program, and education programs featuring tuition advancement and the unique MITRE Institute.

If you have a personal computer or terminal with a telephone connection, explore MITRE's opportunities in Massachusetts on line by calling (617) 271-8000. Use the login name 'mitre' followed by a carriage return.

An Equal Opportunity/Affirmative Action Employer.

BOSTON

Please send your resume to:
David L. Finnegan
The MITRE Corporation
7303 Middlesex Turnpike
Bedford, MA 01730

WASHINGTON

Please send your resume to:
M.C. Mason
The MITRE Corporation
7525 Colshire Drive
McLean, VA 22102



VLSI Implementation of a Neural Network Model

Hans P. Graf, Lawrence D. Jackel, and Wayne E. Hubbard
AT&T Bell Laboratories

Models of neural networks are receiving widespread attention as potential new architectures for computing systems. The models we consider here consist of highly interconnected networks of simple computing elements. A computation is performed collectively by the whole network with the activity distributed over all the computing elements. This collective operation results in a high degree of parallel computation and gives the network the potential to solve complex problems quickly. Neural network models have demonstrated functions such as associative memory, adaptive learning from examples, and combinatorial optimization.

To date, the research concerning neural network models has focused mainly on theoretical studies and computer simulations. However, the real promise for applications of the models lies in specialized hardware, in particular specialized micro-electronic circuits. Simulations of large networks on serial computers are painfully slow, and only with customized hardware can we hope to realize neural network models with speeds fast enough for applications.

Digital accelerators to simulate neural networks are now commercially available, but they are still orders of magnitude slower than what we can achieve by directly fabricating a network with hard-



To obtain the full benefit of neural network algorithms, we need special-purpose hardware. An experimental CMOS VLSI circuit was tested as an associative memory and as a pattern classifier.

ware. Several researchers have built models with discrete electronic components. These implementations help us study properties such as the dynamics of these circuits, but they are too bulky for real applications.

The most promising approach for implementing electronic neural nets is to fabricate special-purpose very-large-scale-integration chips. With today's integration

density, a large number of simple processors can be packed on a single chip together with the necessary interconnections to make a collective computing network. Several groups have initiated experiments with VLSI implementations and demonstrated a few functioning circuits.¹⁻³

Attempts are under way to build neural network models with optics.⁴ The high interconnectivity of the networks makes optics attractive because interconnections can be made optically in three dimensions. The circuit on a microchip is bound to the two dimensions of the chip's surface. However, optical computing technology is still in its infancy and realizations suitable for applications probably lie further in the future.

We describe a complementary metal oxide semiconductor (CMOS) very large scale integrated (VLSI) circuit implementing a connectionist neural network model that consists of an array of 54 simple processors fully interconnected with a programmable connection matrix. This experimental design tests the behavior of a large network of processors integrated on a chip. We can operate the circuit in several different configurations by programming the interconnections between the processors. We made tests with the circuit working as an associative memory and as a pattern classifier. The results were so encouraging that we interfaced the chip to

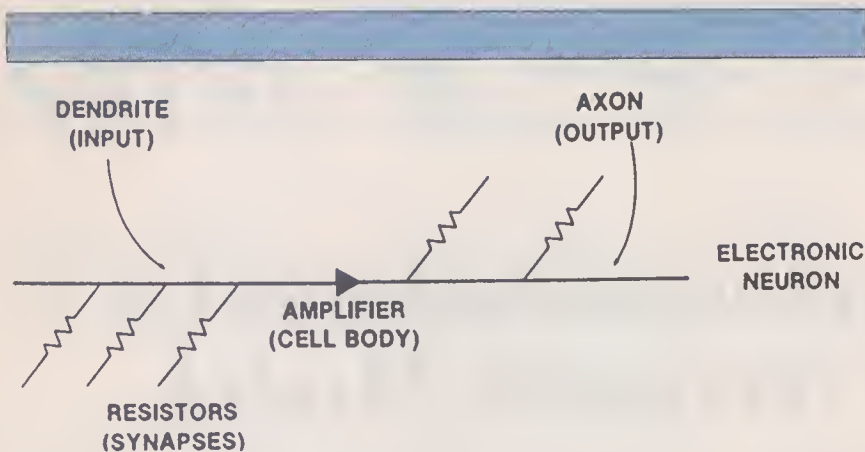


Figure 1. An electronic "neuron."

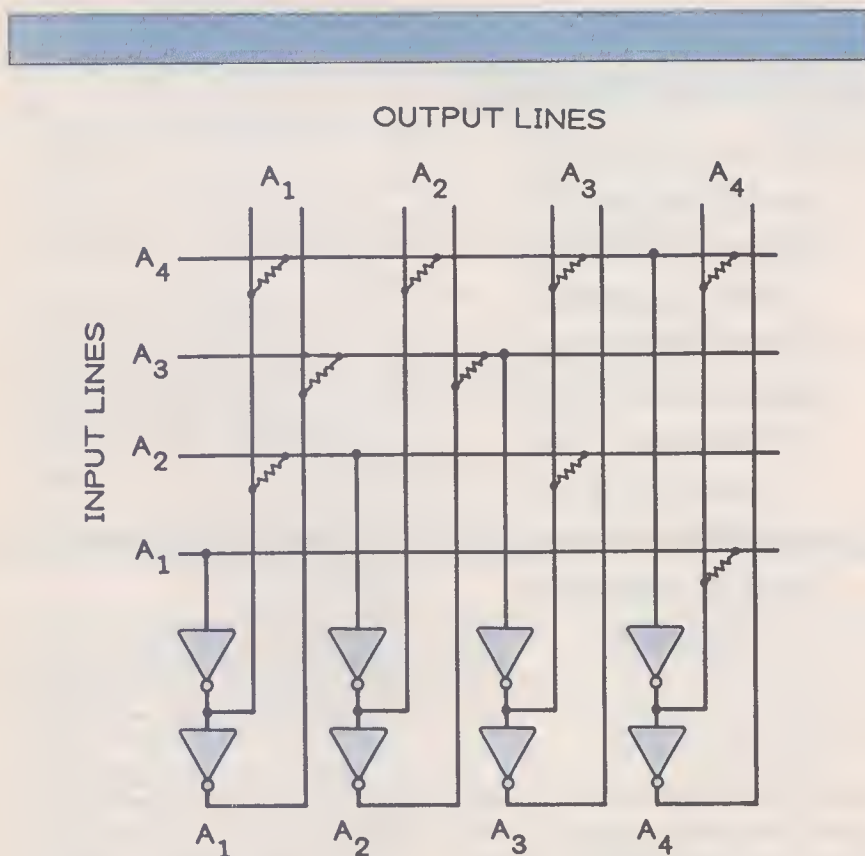


Figure 2. Schematic of the implemented circuit. At each crosspoint of an input and an output line, a resistive connection can be set. All the connections are programmable. The configuration of resistors shown here is just one example. Two inverters are connected in series and work as one amplifier unit. To have the inverted and the noninverted output going into the connection matrix makes it easy to control excitatory and inhibitory connections.

a minicomputer and now use it as a coprocessor in pattern recognition experiments. This mode of operation allows us to test the chip's behavior in a real application and study how pattern recognition algorithms can be mapped into such a network.

Connectionist model

Biological neural networks inspired the models we are implementing in electronics, but we are not attempting to imitate real neurons. Clearly, the models used grossly simplify the biological networks. But even these relatively simple networks have complex dynamics and show interesting collective computing properties. It is this collective computation that we try to exploit by building such networks.

The type of circuit described here is often referred to as a *connectionist model*. In a connectionist model, an individual "neuron" does very little computation, typically just a thresholding of its input signal. The kind of computation performed by the whole network depends on the interconnections among the neurons. Figure 1 shows a possible electronic circuit for such a simple neuron. An amplifier models the cell body, wires replace the input structure (dendrite) and the output structure (axon), and resistors model the synaptic connections between neurons. The amplifier's output voltage replaces the firing rate of a real neuron.

Each of the resistors connects the input line of the amplifier to the output line of another amplifier. Therefore, the state of an amplifier is determined by the states of all the other amplifiers. When the amplifier measures the current flowing into the input line, we can express this as

$$V_{out_j} = f\left(\sum_i I_i\right) \\ = f\left(\sum_i (V_{out_i} - V_{in_i})T_{ij}\right) \quad (1)$$

where V_{in} , V_{out} are the input and output voltages of an amplifier; I_i is the current flowing through one resistor; T_{ij} is the conductance of the resistor connecting amplifier i with amplifier j ; and $f()$ is the transfer function of the amplifier.

Equation 1 shows how the states of the amplifiers, represented by V_{out_j} , determine how much current flows into the input line and therefore determine the state of this amplifier. The output voltage of the amplifier is then given by its transfer characteristics. Its output connects to the input of many other amplifiers and

influences their states. A network of highly interconnected amplifiers forms in this way. It is very difficult to describe the dynamics of such a system for an arbitrary distribution of the resistive connections, but many special cases can be controlled well and exploited for collective computation. Such networks have been demonstrated as functional associative memory and as optimizers.^{5,6}

The implemented circuit well suits work as an associative memory or as a pattern classifier. To have the network perform these functions does not require precise control of the gain of the amplifiers or the ability to fine tune the resistive connections. This makes it possible to design interconnections and amplifiers using only a small area, so many of these components can be integrated on a single chip.

The function described by Equation 1 could be implemented in digital hardware. But this would require a complex multiplier-adder circuit at the input of each neuron, resulting in a large circuit even for a modest number of neurons. The implementation described here uses a mixture of analog and digital CMOS VLSI technology.

Using analog computation, we can achieve a multiplication with a single resistor, and the summing of currents is accomplished "for free" on the input wire of the amplifier. However, designing large integrated analog circuits is a difficult task. There is a strong tendency in signal processing today to avoid analog computation and do everything digitally. Yet the high interconnectivity and relatively low precision needed for the signals in a neural net are well tailored for an analog approach, and the gain in computing power of the network should outweigh the extra design effort.

The circuit

Figure 2 shows a schematic of the implemented circuit. It consists of an array of 54 amplifiers with their inputs and outputs interconnected through a matrix of resistive coupling elements. All of the coupling elements are programmable—a resistive connection can be turned on or off.

Figure 3 shows a photomicrograph of the circuit. Fabricated in CMOS technology with 2.5-micrometer design rules, it contains roughly 75,000 transistors in an area measuring 6.7×6.7 millimeters. By far the largest active area of the chip, almost 90 percent, is used for the program-

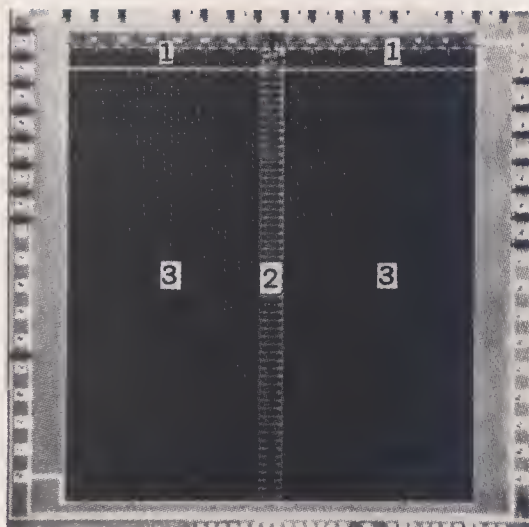


Figure 3. Photomicrograph of the chip. The modules are: 1 = amplifiers, control logic, bit decoder, input bus, and output bus; 2 = decoder to address rows in the interconnection matrix; 3 = interconnection matrix.

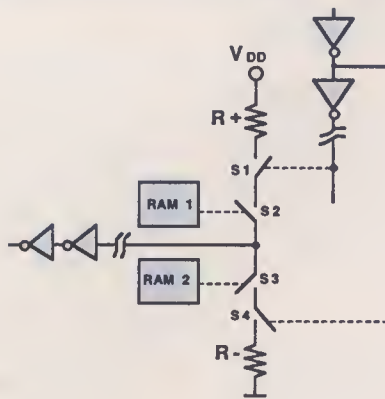


Figure 4. One of the coupling elements connecting the output of an amplifier to the input of another amplifier. S_1 and S_4 are closed when the output of the amplifier controlling them is high (noninverted output high). If a "1" is stored in RAM 1, S_2 is closed and the connection is excitatory. If a "1" is stored in RAM 2, S_3 is closed and the connection is inhibitory. If both RAM cells store a "0," no current flows through this interconnection regardless of the state of the controlling amplifier.

mable coupling network.

In this design the circuit shown in Figure 4 replaces the resistors of Figure 2. The output lines of the amplifiers do not feed current into the input lines; instead they control the switches S_1 and S_4 . This method reduces the amplifier load to just

the capacitance of the output line. For each connection between two amplifiers, two memory cells control switches S_2 and S_3 . The content of these memory cells determines the type of connection. One of three connections can be selected: an excitatory (S_2 enabled), an inhibitory (S_3

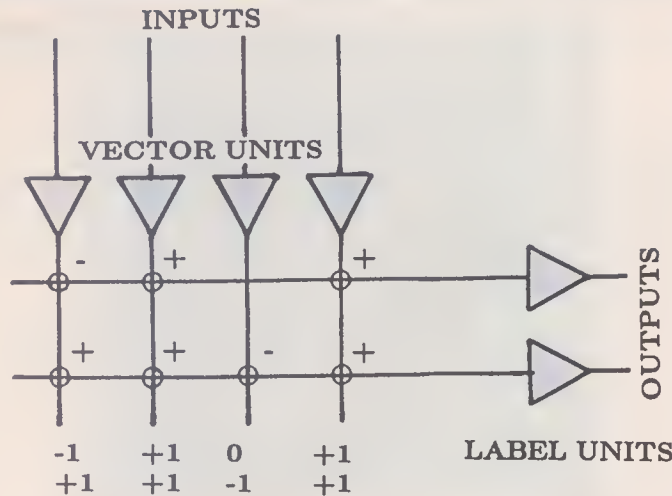


Figure 5. Programming of the interconnections for the pattern classifier circuit. The "+" and the "-" mark excitatory and inhibitory connections, respectively. Below the schematic the stored vectors are shown.

enabled), or an open (both disabled) state. The network contains a total of 2916 such coupling elements. Notice that when an amplifier output is low, the connection turns off completely and no current flows in either direction.

The voltage of an input line of an amplifier is determined by the sum of the currents flowing into the node. This voltage adjusts to a value where the total current is zero. Since the input impedance of the amplifiers is very high, this leads to

$$\sum_{i=0}^{i=N} I_{ij} = \sum_{i=0}^{i=N} \frac{\Delta V_{ij}}{R_{ij}} = 0 \quad (2)$$

where I_{ij} is the current flowing through a resistor of the coupling element controlled by amplifier i ; ΔV_{ij} is the voltage difference across a resistor, ($V_{in_j} - V_{DD}$, $V_{in_j} - V_{SS}$); and R_{ij} is the resistor, (R_+ , R_-).

Thus, the voltage V_{in_j} is an analog measure of the sum of the contributions of all the amplifiers connected to input line j . The amplifiers used have a high gain and work essentially as threshold elements, with the switching threshold about halfway between the ground and the supply voltage. Analog computation is used only within the connection matrix; input and output data and all the control signals are digital.

Data input and output are transferred through a register where one memory cell

is connected to each amplifier. The input data are first loaded into this register. From there they can be loaded into the memory cells of the connection matrix, or used to initialize the circuit. Initialization of the circuit is done by charging the network with the voltage levels corresponding to the components of the input vector. The amplifiers are turned off during this initialization cycle. For the computation, the amplifiers are turned on and the network evolves to a stable state without any external control or synchronization between the amplifiers. After the circuit has reached a stable state, the output voltage of each amplifier is stored in the register, which can then be read out.

Programming the chip

The circuit's architecture facilitates mapping several different configurations into the network simply by programming the connections between the amplifiers. Figure 5 shows the arrangement of the interconnections for a configuration used to do pattern classification. The amplifiers are divided into two groups: the label units and the vector units. A number of vectors are stored in the circuit, each one along the input line of one label unit. The components of the stored vectors can have the values +1, -1, or 0. An excitatory con-

nection is set for a +1 and an inhibitory connection for a -1.

The input vector is presented on the inputs of the vector units. Its components can have the values +1 or 0. Wherever a +1 in the input vector is set, current can be injected or drawn from the input line of a label unit depending on the type of the connection. As described by Equation 2, the condition for a stable state is that the total current flowing into an input line equals 0. If the input voltage is above the threshold of the amplifier, this label unit turns on; otherwise, it remains off. Whether a label turns on or not is described by Equation 3:

$$\sum_{i=0}^{i=N} \frac{v_i \mu_i}{R_i} = \begin{cases} > 0: V_{out} = \text{high} \\ < 0: V_{out} = \text{low} \end{cases} \quad (3)$$

where v_i denotes components of the input vector, (+1, 0); μ_i denotes components of the stored vectors, (-1, 0, +1); and R_i is the resistance of the connections, (R_- , R_+).

The input vector is compared in parallel with all the stored vectors and an inner product between the input vector and the stored vectors is evaluated. All the stored vectors closely resembling the input vector turn on their label units.* A +1 in the input vector at the position of a +1 in the stored vector gives a positive contribution to the sum, while a -1 in the stored vector in this position gives a negative contribution. R_+ is about six times larger than R_- . Therefore, a mismatch (a +1 in the input in the position of -1 in the stored vector) counts six times as much as a match. This ratio of R_+ to R_- has no great significance; it simply reflects the ratio of the resistance of the p-channel and the n-channel transistors of the CMOS circuit. For the applications described later, it is convenient to have the inhibition stronger than the excitation, but we could obtain the same effect by using multiple connections.

By using a few amplifiers as bias units, we can shift the threshold of a label unit. We can program the connections between the bias units and the label units to set different thresholds for each stored vector. The right-hand side of Equation 3 is then replaced by the bias value.

*The classification properties of this circuit correspond to those of a single-layer perceptron; it can discriminate between linearly separable patterns. Since we are dealing with binary vectors, the configuration space consists of the corners of an n -dimensional hypercube. Decision regions of any shape can be built in this space as the sum of linearly separable regions.

The number of stored vectors and their lengths are limited by the number of amplifiers in the circuit. Each component of the input vector uses one amplifier, and each label unit needs one. Therefore, the number of bits in a vector plus the number of stored vectors must be smaller than 54. Usually, a few amplifiers are used for biases, reducing this number to approximately 50.

This arrangement uses only the connections between the outputs of the vector units and the inputs of the label units, while the largest part of all the connections in the matrix remains idle. We can achieve a more efficient use of the circuit by using the register cells as the vector units. The arrangement described uses these cells only as intermediate storage for the input vector, but they can also charge the matrix to its initial condition. All the amplifiers then work as label units. Up to 54 vectors, each 54 bits long, can be stored in the network and are compared in parallel to the input vector. To read out the result, the register cells must switch quickly from writing the input vector to reading the result without feeding the result back into the connection matrix. This requires a precisely timed pulse controlling the length of time the amplifiers are turned on. This mode of operation is used for feature extraction from images (see the section "Examples of applications").

Adding inhibitory connections between the label units and connecting the outputs of the label units to the inputs of the vector units can yield an associative memory. This arrangement is shown schematically in Figure 6. In this circuit, vectors with components +1 and 0 are stored. In addition to the connections along the input lines of the label units described above, there are inhibitory connections between all the label units. Each label unit inhibits all the other label units, but not itself. For the connections between the label outputs and the vector inputs, the same vector is placed along the output line of a label unit stored along its input line. For a +1 in the vector, an excitatory connection is set, and for a 0, an inhibitory connection is set.

The circuit is initialized with all input lines of the label units discharged to ground. The input vector is given on the input lines of the vector units. Wherever a +1 in the input vector and in a stored vector occupy the same position, current is injected into the input line of a label unit. The speed at which an input line of a label unit changes state depends on the inner

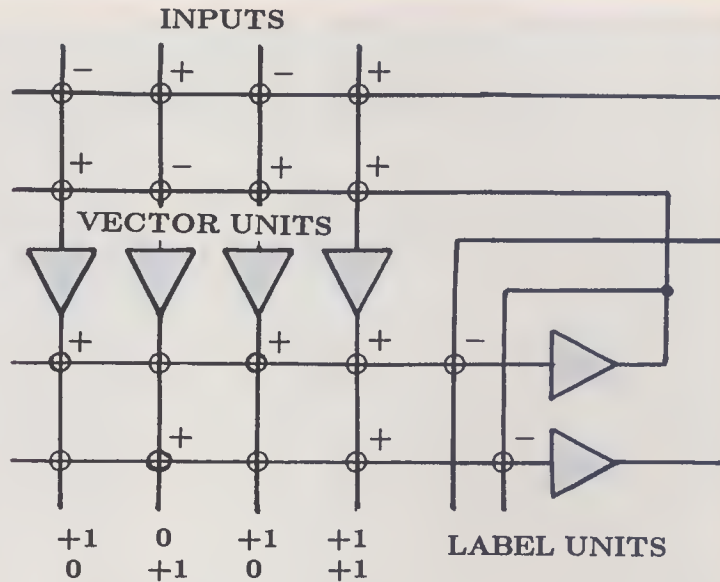


Figure 6. The interconnections set for an associative memory circuit.

product of the input vector and a stored vector. That is,

$$\frac{dV_{in}}{dt} = \frac{1}{CR_+} (V_{DD} - V_{in}) \sum_{i=1}^N v_i \mu_i \quad (4)$$

where V_{in} is the voltage of the input line; C is the capacitance of the input node of a label; R_+ is the resistance of the excitatory connection; v_i denotes the components of the input vector, (0, +1); and μ_i denotes the components of the stored vector, (0, +1).

The stored vector that has the largest inner product with the input vector will turn on its label unit first. As one label unit turns on, it inhibits all the other label units. If this inhibition is strong enough, no other label unit will turn on.

As mentioned above, an inhibition is six times stronger than an excitation. This limits the allowed mutual overlap (number of common 1's) of two stored vectors to five bits; biases are used to circumvent this limitation. The label unit that comes on first generates the vector stored along its output wire at the inputs of the vector units. In this way, in a stable state one label unit is on and the vector connected to that label appears at the outputs of the vector units.

This circuit performs a minimum distance classification. The input vector is

compared with all the stored vectors in parallel, and the circuit determines which of them most closely matches the input vector (common 1's are counted).

Extensive tests with 10 stable states, each 40 bits long, programmed into the associative memory circuit showed that the circuit converges to a stable state in 50 to 600 nanoseconds.² Associative recall performed reliably, and we observed only the stable states programmed into the circuit; we saw no spurious stable states. The convergence time depends on how closely the input vector resembles the stored states. Additional tests run with up to 20 stable states, each 30 bits long, programmed into this circuit obtained similar results.

In this circuit the stored data are represented locally in the interconnection matrix. This means that a stored bit can be localized at one or two of the interconnections. In contrast, other circuits based on neural network models use a distributed representation of the data (e.g., the outer product of a stored vector determines the distribution of the connections).

The great interest in associative memory circuits implemented with neural network models has resulted in the description of several programming methods.^{5,7} Experiments conducted with this type of associative memory soon made clear the far

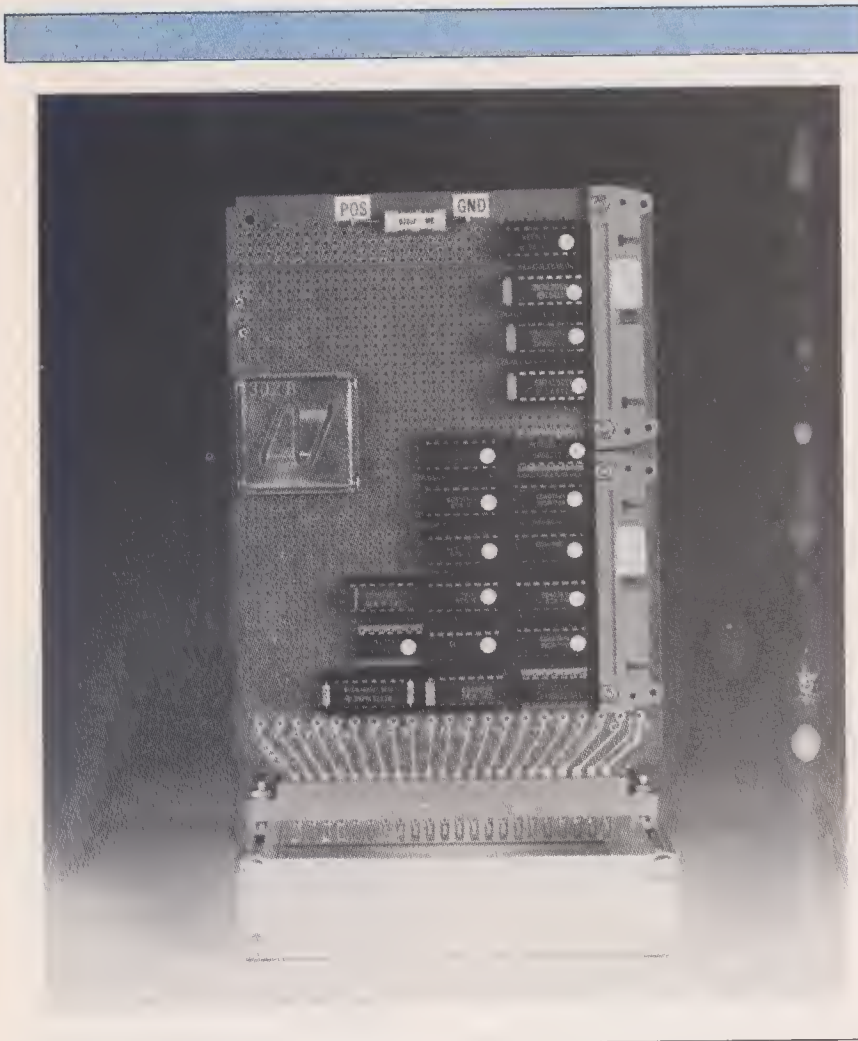


Figure 7. The interface connecting the chip to a minicomputer. The network chip is on the left side of the board. The other chips control the data exchange with the computer.

superior results given by local representation of data. With distributed representation of data, the different stored vectors influence each other because they share the same interconnections. This leads to unwanted spurious stable states and to erroneous recalls of vectors that do not correspond to the nearest neighbor.

These problems do not exist in the circuit we describe here. We observed reliable recall of the nearest neighbor and no unwanted stable states. Moreover, the storage density is considerably higher with this circuit. With a distributed representation of the data, it would hardly be possible to get 20 arbitrary stable states in this circuit. But the arrangement described here still does not use the interconnections of the chip efficiently. None of the connec-

tions among the vector units are used. In the case of 10 stored vectors, this means that less than one third of all the connections are used.

With the present chip we do not see a way to further increase storage efficiency. However, a design optimized for this new associative memory circuit can achieve 100 percent storage density.² This means that one RAM cell stores one bit. In contrast, associative memory circuits with distributed representation of data typically require five or more RAM cells to store one bit. High storage efficiency is crucial for an electronic implementation. Other researchers have recently proposed circuits similar to the associative memory we have described.^{8,9}

We can configure the circuit to move

through a sequence of vectors by placing a vector along the output line of a label that differs from the vector along its input line. Mixing connections of two vectors can stabilize the circuit at a vector and make it sensitive to the next vector in the sequence only. When the input lies within a set range of this vector, the circuit will move to the next stable state. Otherwise, it just stays in its present state. Other versions of programming techniques allow for omissions and branches in a parsed sequence of vectors.¹⁰

Examples of applications

In order to evaluate the behavior of the circuit in an application, we used it in a character recognition experiment. An interface connected the chip to a minicomputer to handle the transfers of the large amounts of data characteristic of image processing requests.

Figure 7 shows the board with the network chip and some additional off-the-shelf integrated circuits to control data flow. Data transfers can be made directly from the minicomputer's memory to the chip at a rate between one and two megabytes per second. This rate is limited by the interface and the minicomputer and not by the chip. One complete processing cycle, which includes loading the input vector, accomplishing a computation with the circuit, and reading back the result into the computer, requires approximately 25 clock cycles or 25 to 50 microseconds. Most of the time is required for reading the data in and out. The processing in the circuit requires only one clock cycle.

The whole process of recognizing a character proceeds as follows: A handwritten character is read with a camera, digitized, and then normalized in size to fill a 128×128 -pixel frame. The image is then coarse-blocked into a 16×16 -pixel binary image. After this, the character is *skeletonized*—the width of the lines is reduced to one pixel—and the skeletonized picture is searched for a number of geometrical features. The positions of these features are compared to a training set and a best match with one of these training characters is determined. Of this whole process, the line thinning and the feature extraction have been mapped onto the chip; the minicomputer accomplishes the remaining operations.

Figure 8 shows an example of a result of the line thinning operation. The chip is

used in the configuration of Figure 5 for this task. Stored in the circuit are 20 different vectors, each representing a 5×5 -pixel area. The input vector to the chip is a 5×5 -pixel region of the character image, which we will refer to as a *card*. Based on which label units are on after a run, a decision is made whether or not to delete the center pixel of this card from the image. For the next run the card is shifted by one row or column of pixels on the image; this new card is used as the input vector. In this way the whole picture is scanned. For each pixel position, a processing cycle is accomplished with the appropriate card as the input vector. For each pixel in the image, a decision has to be made whether that pixel makes the line fat or whether its presence is crucial to keeping intact the connectivity of the character.

Figure 9 shows an example of a vector stored in the circuit to make one such decision. If the label connected to this vector turns on, then the center pixel is part of a diagonal line or a corner and that line is at least two pixels wide. In this situation, the pixel can be deleted without destroying the connectivity of the character. The 20 vectors stored on the chip analyze the neighborhood of each pixel for all the configurations that allow its deletion. All the computation needed to decide whether a pixel can be deleted is done in one clock cycle. A total line-thinning operation requires about three scans across the whole character, depending on the width of the lines because only boundary pixels are deleted.

Line thinning is an important step in machine vision, not only in character recognition, but also in tasks like fingerprint analysis and inspection of manufactured parts. Many algorithms have been developed to handle this problem.¹¹ The algorithm implemented with the network circuit does not differ fundamentally from other pixel-based algorithms. The stored vectors facilitate making the same type of tests as those formulated in the other algorithms with a set of Boolean functions. Most other algorithms base their decision on a 3×3 area around the pixel under test, since the test of larger areas becomes very time consuming. However, with the network it is not a problem to analyze the larger 5×5 pixel area, since it still takes only one clock cycle. Using a larger area makes the algorithm more robust, and it supports integrating some smoothing to enhance the thinning operation.

To accomplish the extraction of geometrical features, 40 vectors are stored in the

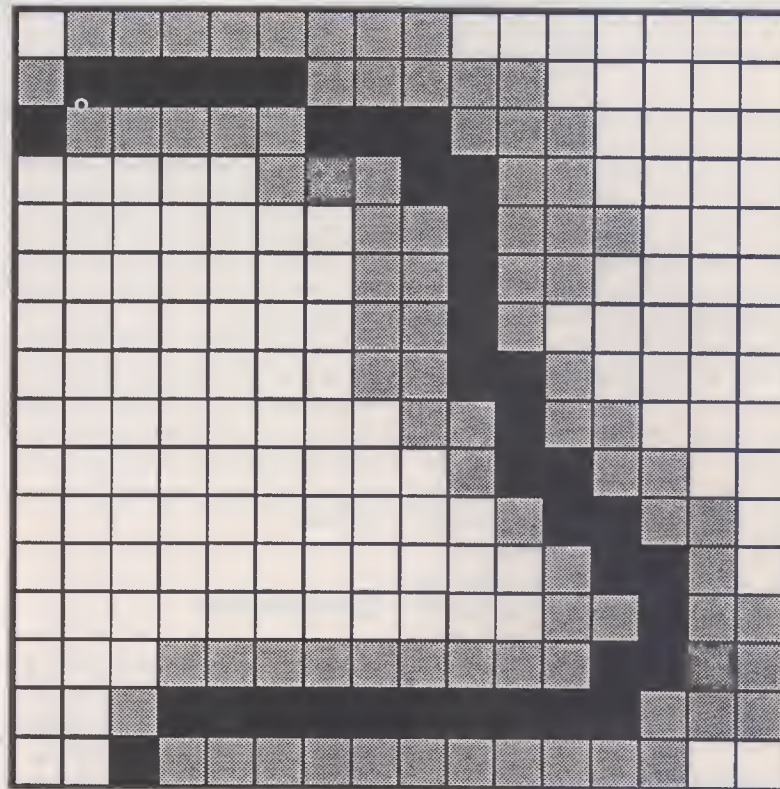
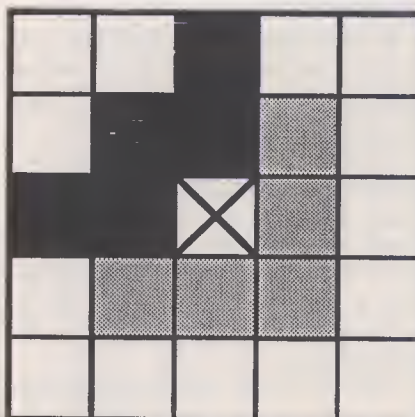


Figure 8. The result of a line-thinning operation on a handwritten "3." The gray area represents the original character, and the black area is the portion that remains after three thinning scans.



00+000++-0+++00---000000

Figure 9. One of the kernels stored in the chip for the line-thinning operation. If the label connected to this vector turns on, the center pixel is erased in the image. The black pixels are coded as excitatory connections (+) and the gray pixels as inhibitory connections (-). The bias is set to -4. The label turns on whenever five black pixels in the image correspond to the black pixels in this kernel and no black pixel in the image is at a position of a gray pixel in the kernel. Then, the center pixel is a part of the boundary of a thick diagonal line or a corner and can be deleted.

circuit. Each one looks for a feature like a straight line, an endpoint of a line, a corner, crossing lines, or an arc. Whenever one of these features is present in the image, the label of this feature-vector turns on.

The image is scanned sequentially, as described for the line-thinning operation. For every pixel position, a 7×7 pixel neighborhood is searched in parallel for the 40 different features. Such a scan results in 40 feature maps that indicate where particular features occur. To compress this large amount of data, the resolution of the feature maps is then reduced from 16×16 to 3×3 positions. These feature maps are compared with reference characters and the best match is chosen. Currently, the minicomputer accomplishes this last matching operation, but a way to map this final operation onto the network circuit is under development.

The successful recognition rate for hand-written digits is approximately 90 percent. We assume that we can improve this rate considerably by using finer reso-

lution for the character images as well as the feature maps. We now use only local geometrical features of a maximum size of 7×7 pixels. We can expect better results when large features such as long strokes or large circles are generated first from these local features. However, the main purpose of this experiment was to test the circuit's performance and to gain experience programming different algorithms into the network. We have not yet pushed the analysis to find the highest recognition rate.

Character recognition based on feature extraction is more robust against distortions than simple template-matching algorithms that compare the pixel positions of the whole character to reference data. Feature extraction is one of the most versatile functions of low-level machine vision and can be applied to many problems.

Discussion

We have tested the circuit with several programmed algorithms and conclude that a network like this one is reliable and robust enough for applications. We designed this circuit to study the behavior of a large analog network and did not specialize it for any particular application. Since input and output of data limit the processing, we must optimize the I/O structure in new designs for particular applications.

The analog portion of the circuit represents a very powerful processor. With 50 stored vectors, each 50 bits long, the chip completes a processing cycle in less than one microsecond. This means that the circuit evaluates 50 inner products between two 50-bit vectors per microsecond. On a standard microprocessor this computation would require more than one hundred instructions.

A new chip being fabricated was designed specifically for image processing. The input vector is shifted in a shift register along the inputs of the amplifiers. In this way a new input vector is ready for a computing cycle at each clock cycle when a card of 8×12 pixels is scanned over an image. It is possible to store 46 vectors, each 96 bits long, and simulations indicate that a complete program cycle can be accomplished in approximately 100 nanoseconds.

This circuit will evaluate several hundred million inner products between two 96-bit vectors per second. For example, the circuit can do the line-thinning operation

described above at a rate of a few hundred microseconds per character compared to the few hundred milliseconds per character a standard computer takes.¹⁰ (With character sizes of 32×32 pixels, each scan requires 1024 cycles if the whole area is scanned; about three scans are required.)

This new design is fabricated with the same conservative 2.5 micrometer CMOS process. Switching to smaller design rules will allow packing considerably more circuitry on a single chip. Also, since this network is implemented with a standard digital fabrication process, it can be combined easily with other memory and processor modules on the same chip to enhance its versatility.

A system concept with a layered structure is also under development. In this scheme a layer of network processors is followed by a memory module, with several of these units stacked in series. In an application like character recognition, the different tasks such as line thinning or feature extraction are then done in different processor layers. Each layer of processors inputs the data from the memory module below it and outputs its results in the memory module above it.

The data flows mainly in one direction, from the raw data at the input of the lowest layer to the output layer that does the pattern identification. However, communication is also provided in the opposite direction, so that results from a higher layer can determine the operation performed in a lower level. This feature facilitates scanning certain areas in the image with a different resolution, or scanning the image for different features when the results in the higher levels are ambiguous. We need additional research on the mapping of different algorithms into the network, and how to format the data optimally to feed from one network into the next with minimal intermediate reformatting.

The network described provides a flexible tool because it can evaluate Boolean expressions and arithmetic equations. Methods of statistical as well as structural pattern recognition can be mapped into the chip. Ideas from the artificial intelligence community on bit-mapped classifiers¹² suggest that expert systems could be made from this network. With all these elements at hand, this network looks promising for building powerful recognition systems. □

ADVANCED COMPUTER ARCHITECTURE

The Microelectronics and Computer Technology Corporation, MCC, a consortium of U.S. companies conducting advanced research in the areas of semiconductor packaging, VLSI CAD systems, software technology and advanced computer architecture is currently seeking researchers with strong backgrounds in any of the following areas:

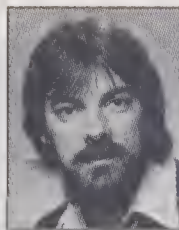
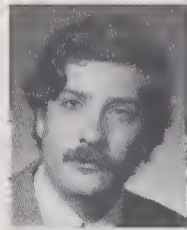
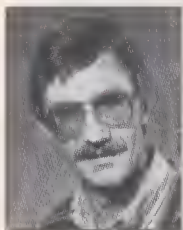
- Object-oriented or functional programming languages
- Distributed database or operating systems
- Object-oriented applications

Candidates should have a MS or PhD in computer science or a related area and experience with object-oriented or distributed systems.

You will find MCC the most exciting research organization in America. Located in AUSTIN, Texas, one of America's most livable cities, we offer a unique career opportunity with excellent compensation and relocation benefits. Find out more about joining this advanced research team. Send your resume to: Vice President, Human Resources, MCC, P.O. Box 200195, Austin, Texas 78720.

An Equal Opportunity Employer

MCC



Hans Peter Graf is a member of the technical staff at AT&T Bell Laboratories. He is doing research on collective computing systems and has designed several CMOS chips implementing connectionist neural network models. He has also been involved in the development of microfabrication processes of resistive connections with submicron dimensions for neural network circuits.

Graf received the diploma in physics in 1976 and the PhD in 1981 from the Swiss Federal Institute of Technology in Zurich, Switzerland.

Lawrence D. Jackel has been at AT&T Bell Laboratories since 1975 and is now head of the Device Structures Research Department. His current research interests include the physics, fabrication, and application of devices with nanometer scale features and electronic neural networks.

Jackel received the BA degree in physics from Brandeis University in 1969 and the PhD degree from Cornell University in 1976. His thesis research was in the physics and applications of superconducting weak links.

Wayne E. Hubbard is a member of the technical staff at AT&T Bell Laboratories, where he has been involved in such diverse projects as optical recording, vapor phase epitaxial growth systems, and currently, neural networks.

Hubbard began his career in electronics and computer science in 1969 at RCA, then served six years with the US Navy Tactical Data Systems. From 1976 until joining Bell Labs in 1982, he worked for Digital Equipment Corp. as a district support engineer.

Readers may write to the authors at AT&T Bell Laboratories, Room 4G320, Holmdel, NJ 07733.

Acknowledgments

We acknowledge many helpful discussions with Henry Baird, John Denker, Richard Howard, and Sara Solla. We designed the chips with the MULGA design tools, and acknowledge the support of Brian Ackland and Robert Clark in using these tools.

References

1. M.A. Sivilotti, M.R. Emerling, and C.A. Mead, "VLSI Architectures for Implementation of Neural Networks," *Proc. Conf. Neural Networks for Computing*, J.S. Denker, ed., 1986, American Institute of Physics Conf. Proc. 151, pp. 408-413.
2. H.P. Graf and P.G.N. deVegvar, "A CMOS Implementation of a Neural Network Model," *Advanced Research in VLSI, Proc. Stanford Conf. 1987*, P. Losleben, ed., MIT Press, Cambridge, Mass., 1987, pp. 351-367.
3. J.P. Sage, K. Thompson, and R.S. Withers, "An Artificial Neural Network Integrated Circuit Based on MNOS/CCD Principles," *Proc. Conf. Neural Networks for Computing*, 1986, J.S. Denker, ed., American Institute of Physics Conf. Proc. 151, pp. 381-385.
4. Y.S. Abu-Mostafa and D. Psaltis, "Optical Neural Computers," *Scientific American*, March 1987, pp. 88-95.
5. J.J. Hopfield, "Neurons with Graded Response Have Collective Computational Properties Like Those of Two State Neurons," *Proc. Academy of Science USA*, Vol. 81, 1984, pp. 3088-3092.
6. D.W. Tank and J.J. Hopfield, "Simple 'Neural' Optimization Networks: An A/D Converter, Signal Decision Circuit, and Linear Programming Circuit," *IEEE Trans. Circuits and Systems*, Vol. CAS-33, 1986, pp.533-541.
7. J.S. Denker, "Neural Network Models of Learning and Adaptation," *Physica D*, Vol. 22D, 1986, pp. 216-232.
8. J. Moody, "Perspectives on Associative Memories," *Proc. IEEE First Int'l Conf. Neural Networks*, M. Caudill and C. Butler, eds., IEEE, 1987.
9. R.P. Lippmann, "An Introduction to Computing with Neural Nets," *IEEE Acoustics, Speech, and Signal Processing*, Vol. 4, 1987, pp. 4-22.
10. P.G.N. deVegvar and H.P. Graf, "Studies of Associative Memory and Sequence Analyzer Circuits on a Programmable Neural Network Chip," to be published. Contact Graf for preprint copies.
11. N.J. Naccache and R. Shingal, "SPTA: A Proposed Algorithm for Thinning Binary Patterns," *IEEE Trans. Systems, Man, and Cybernetics*, Vol. SMC14, 1984, pp. 409-418.
12. P.W. Frey, "A Bit-Mapped Classifier," *Byte*, Nov. 1986, pp. 161-172.

**Late Magazines?
No Magazines?
Membership
Status Problems?
No Answers
To Your
Complaints?**

**Let your
Computer
Society
Ombudsman
cut
through
the red
tape
for you.**



Computer Society
Ombudsman
IEEE Computer Society
10662 Los Vaqueros Circle
Los Alamitos, CA 90720

ADVANCE PROGRAM

1988 IEEE Computer Society Symposium on Security and Privacy

April 18-20, 1988 • The Claremont Hotel • Oakland, California

*Sponsored by the IEEE Technical Committee on Security and Privacy
In cooperation with the International Association of Cryptologic Research*

Monday, April 18

- 8:45-9:00 am Welcome, Introduction
- 9:00-10:30 am **Formal Security Models**
Dick Kemmerer, Chair
The Algebra of Security: *John McLean*
Concerning "Modeling" of Computer Security:
D. Elliott Bell
Security Specifications: *Jeremy Jacob*
- 10:30-11:00 am Break
- 11:00 am-12:00 noon **Panel: Security Models**
Dick Kemmerer, Chair
D. Elliott Bell, John McLean
- 12:00 noon-1:30 pm Lunch
- 1:30-3:00 pm **Security in Distributed Systems**
Dan Nessett, Chair
Blacker: Security for the DDN: *TBA*
Examples of AI Security Engineering Trades:
Clark Weissman
A Secure Distributed Operating System:
Thomas A. Casey, Jr and Stephen T. Vinter
Extended Discretionary Access Controls:
Stephen T. Vinter
- 3:00-3:30 pm Break
- 3:30-5:00 pm **Emerging Issues**
Tom Berson, Chair
A Fault Tolerance Solution to Computer
Viruses: *Mark K. Joseph and Algirdas*
Avizienis
A Prototype IDDES—A Real-Time Intrusion
Detection Expert System: *Teresa F. Lunt and*
R. Jagannathan
A Stab at an A2 Criteria: *Mario Tinto and*
George Jelen
- 5:30-7:00 pm Reception

Tuesday, April 19

- 9:00-10:30 am **Data Base Security: ASD-Views**
Teresa Lunt, Chair
Views as the Security Objects in a Multilevel
Secure Relational Database Management
System: *Jackson Wilson*
ASD-Views: *Christi Garvey and Amy Wu*
Inference Aggregation Detection in Database
Management Systems: *Thomas H. Hinke*
- 10:30-11:00 am Break
- 11:00 am-12:00 noon **Analysis of Secure Systems**
Debbie Cooper, Chair
A Bandwidth Computation Model for Covert
Storage Channels and its Applications: *Chii-*
Ren Tsai and Virgil D. Gligor
Reasoning About Knowledge in Multilevel
Secure Distributed Systems: *Janice Glasgow*
and *Glenn H. MacEwen*

12:00 noon-1:30 pm Lunch

- 1:30-3:00 pm **Applying and Implementing Integrity Models**
Steve Lipner, Chair
Implementing Commercial Data Integrity with
Secure Capabilities: *Paul A. Karger*
Using Mandatory Integrity to Enforce "Com-
mercial" Security: *Theodore M. P. Lee*
The Trusted Path Between SMITE and the User:
Simon Wiseman
- 3:00-3:30 pm Break
- 3:30-4:00 pm **Cascaded Authentication**
Karen R. Sollins
- 4:00-5:00 pm **Panel: Individual Privacy—Technical, Social,
and Legal Issues**
Maria Pozzo, Chair
Mary Karen Dahl, William H. Murray, Marc
Rottenberg, Christine Borgman, Deborah
Estrin

Wednesday, April 20

- 9:00-10:30 am **Verification**
Terry Vickers Benzel, Chair
Reverification of a Microprocessor: *Stephen*
D. Crocker, Eve Cohen, Sue Landauer,
Hilarie Orman
Noninterference and the Composability of
Security Properties: *D. McCullough*
A Formal Specification and Verification
Method for the Prevention of Denial of
Service: *Che-Fn Yu and Virgil D. Gligor*
- 10:30-11:00 am Break
- 11:00 am-12:00 noon **Models of Secure Distributed Systems**
Earl Boebert, Chair
An Axiomatic Basis of Trust in Distributed
Systems: *V. Ragan*
Security Policy Modeling for the Next Genera-
tion Packet Switch: *D. Elliott Bell*
- 12:00 noon-1:30 pm Lunch
- 1:30-3:00 pm **Secure Data Bases**
Paul Cudney, Chair
The SeaView Security Model: *Dorothy E.*
Denning, Teresa F. Lunt, Roger R. Schell,
Mark Heckman, William R. Shockley
Design for a Near-Term Implementation of the
SeaView Multilevel Database System: *Teresa*
F. Lunt, Roger R. Schell, Mark Heckman,
William R. Shockley
Security and Inference in Multilevel Database
and Knowledge-Base Systems: *Matthew*
Morgenstern

CONFERENCE COMMITTEE

David Bailey
General Chairman
Steve Lipner
Program Co-Chairman

Terry Vickers Benzel
Vice-Chair
Tom Berson
Program Co-Chairman

ADVANCE PROGRAM

1988 IEEE Computer Society Symposium on Security and Privacy

April 18-20, 1988 • The Claremont Hotel • Oakland, California

Claremont Resort Registration • Register Now!

Reservations must be received by the Claremont Resort by March 16, 1988. Reservations received after March 16, 1988 will be accepted on a space available basis. We cannot guarantee the confirmed group rate after March 16.

Complete this form and return to:

The Claremont Resort
P.O. Box 23363
Ashby & Domingo Avenues
Oakland, CA 94623
(415) 843-7926

Please check accommodations and price desired:

- ☐ Single occupancy: \$85.00
☐ Double occupancy: \$97.00

Arrival Date _____ Departure Date _____
☐ am ☐ pm ☐ am ☐ pm

All major credit cards accepted: ☐ VISA ☐ MasterCard ☐ American Express

Card No. _____ Exp. Date _____ Signature _____

Name _____ Company _____

Address _____ City/State/Zip/Country _____

To confirm a reservation, a one room's night advance deposit is required. The above rates do not include the 8% City & State tax.

Advance Registration • Register Now!

Complete this form and return to:

Security and Privacy c/o Terry Vickers Benz
The Mitre Corporation
Burlington Road
Bedford, MA 01730
(617) 271-3295

Conference Registration

Advance (prior to 3/18/88): ☐ Member \$220 ☐ Non-Member \$280 ☐ Student \$50
Late (after 3/18/88): ☐ Member \$260 ☐ Non-Member \$340 ☐ Student \$50

Conference registration includes one copy of the proceedings. Student registrants must be IEEE Student Members, not employed full time. Student registration does not include the proceedings. **Advance registration is mandatory. There will be no on-site registration. Attendance is limited to 300 persons. Requests for refunds must be received in writing no later than March 18, 1988.**

Please answer the following questions.

Are you an author of a submitted paper? ☐ YES ☐ NO

Are you a full-time registered student? ☐ YES ☐ NO

Have you participated in any research, development, or evaluation project in computer security? ☐ YES ☐ NO

If yes, please name the project. _____

Can you discuss the results of the project at this symposium (subject to the constraints of classification and proprietary information)? YES NO

Name _____ Company _____ Mail Stop _____

Street Address _____ City/State/Zip/Country _____

IEEE or IEEE Computer Society Member Number _____

Make check (US Dollars) payable to "IEEE Symposium on Security and Privacy."



THE INSTITUTE OF ELECTRICAL
AND ELECTRONICS ENGINEERS, INC.

Computing Motion Using Analog and Binary Resistive Networks

James Hutchinson*
Jet Propulsion Laboratory

Christof Koch, Jin Luo, and Carver Mead
California Institute of Technology

To us, and to other biological organisms, vision seems effortless. We open our eyes and we “see” the world in all its color, brightness, and movement. Flies, frogs, cats, and humans can all equally well perceive a rapidly changing environment and act on it. Yet, we have great difficulties when trying to endow our machines with similar abilities. In this article, we describe recent developments in the theory of early vision that led from the formulation of the motion problem as an ill-posed one to its solution by minimizing certain “cost” functions. These cost or energy functions can be mapped onto simple analog and digital resistive networks. Thus, we can compute the optical flow by injecting currents into resistive networks and recording the resulting stationary voltage distribution at each node. These networks, which we implemented in complementary metal oxide semiconductor (CMOS) very large scale integrated (VLSI) circuits, represent plausible candidates for biological vision systems.

Motion

The movement of objects relative to eyes or cameras serves as an important

We can compute optical flow by injecting currents into resistive networks and recording the stationary voltage distribution at each node.

source of information for many tasks. We need motion to track objects and to determine whether an object is approaching or receding. Relative motion contains information regarding the three-dimensional structure of objects and allows biological organisms to navigate quickly and efficiently through the environment.

There exist two basic methods for computing motion. Intensity-based schemes

rely on spatial and temporal gradients of the image intensity to compute the speed and the direction in which each point in the image moves. The output is a velocity or motion vector field covering the entire image. The second method is based on the identification of special features in the image, called *tokens*, which are then matched from image to image. This method relies on the unambiguous identification of the tokens—for instance, corners—in each image frame before the matching occurs and only yields a velocity vector at the sparse token locations. Psychophysical evidence suggests that both systems coexist in humans.¹

The principal drawback of all intensity-based schemes lies in the data used—temporal variations in brightness patterns—which give rise to the perceived motion field, the *optical flow*. In general, the optical flow and the underlying velocity field, a purely geometrical concept, differ.² For example, a featureless rotating sphere will not give rise to any optical flow, because the brightness does not appear to change even though the velocity field is non-zero. Conversely, if a shadow moves across the same featureless but now stationary sphere, the optical flow is non-zero although the velocity field is zero. Apart from such situations, the estimated opti-

*Hutchinson is now with Thinking Machines Corp.

cal flow will be very nearly identical to the underlying velocity field, if strong enough gradients exist in the image. In this article, we assume that such strong gradients exist, as they do for most natural scenes, and consider how we can compute the velocity field using simple resistive networks.

Aperture problem. Let us derive an equation relating the change in image brightness to the motion of the image.² We denote the image at time t by $I(x, y, t)$. Let us assume that the brightness of the image is constant over time:

$$\frac{dI}{dt} = 0 \quad (1)$$

This will be true, for instance, if a rigid object translates in space (assuming orthographic projection), but not if it rotates. On the basis of the chain rule of differentiation, Equation 1 transforms into

$$\begin{aligned} \frac{\partial I}{\partial x} \frac{dx}{dt} + \frac{\partial I}{\partial y} \frac{dy}{dt} + \frac{\partial I}{\partial t} &= \\ I_x u + I_y v + I_t &= \nabla I \cdot \mathbf{v} + I_t = 0 \end{aligned} \quad (2)$$

where we define the velocity $\mathbf{v}$ as $(u, v) = (dx/dt, dy/dt)$, and where I_x , I_y , and I_t are the partial derivatives of the brightness I with respect to x , y , and t . Because we assume that we can compute these spatial and temporal image gradients, we now have a single linear equation in two unknowns, u and v , the two components of the velocity vector.

In other words, this equation by itself is not sufficient to determine the velocity field. Figure 1 graphically illustrates this *aperture problem*. Any measuring system with a finite aperture, whether biological or artificial, can only sense the velocity component perpendicular to the edge or along the spatial gradient $(-I_t/|\nabla I|)$. The component of motion perpendicular to the gradient cannot, in principle, be registered. The problem remains unchanged even if we measure these velocity components at many points throughout the image. For each measurement, we recover one equation with two unknowns.

Smoothness assumption. Formally, this problem can be characterized as ill-posed.³ Hadamard introduced this concept to describe problems in mathematical physics that (1) have no solution at all, (2) have no unique solution, or (3) do not depend continuously on the initial data. Inverse problems, such as computer tomography, represent ill-posed prob-

Figure 1. The aperture problem of motion. Any system with finite aperture, whether of biological or artificial origin, can only measure the velocity component $-I_t/|\nabla I|$ along the spatial gradient ∇I . Motion perpendicular to the gradient will not be visible, except by tracking salient features in the image.¹

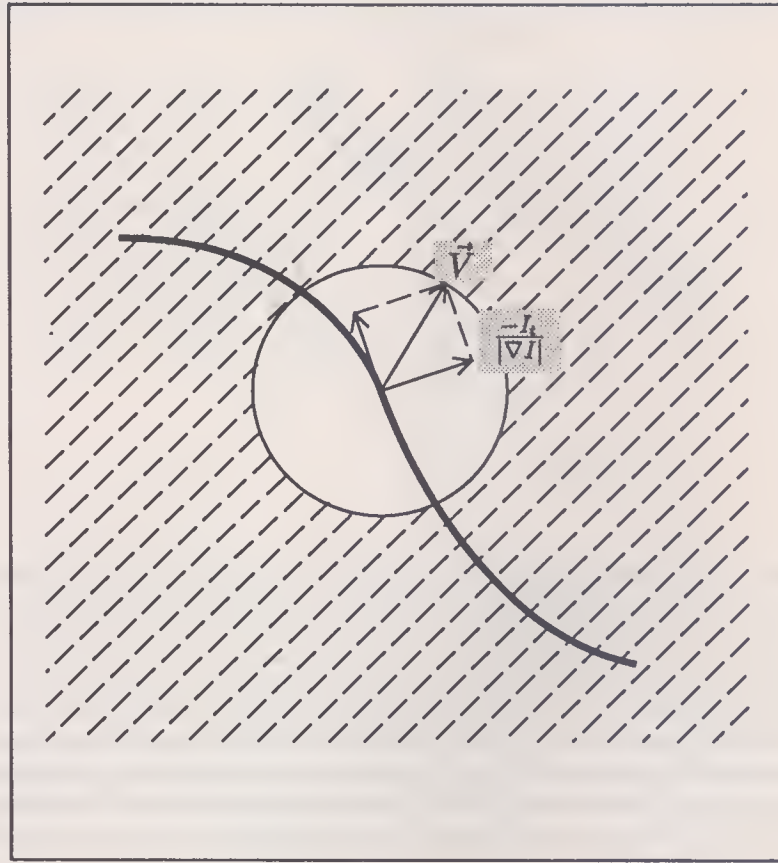
lems. All problems in *early vision*, which we define as the set of processes that recover the properties of the visible three-dimensional surfaces from the two-dimensional intensity arrays on retinæ or cameras, are ill-posed. For example, binocular stereo and interpolating surfaces from sparse and noisy data are ill-posed, because in the former many and in the latter infinitely many solutions exist.

How can we make these problems well-posed, with unique solutions depending continuously on the data? One method of "regularizing" ill-posed problems involves restricting the class of admissible solutions by imposing appropriate constraints.³ Applying this method to

motion, we argue that, in general, objects are smooth—except at isolated discontinuities—undergoing smooth movements. Thus, in general, neighboring points in the world will have similar velocities. The projected velocity field should reflect this fact. We therefore impose on the velocity field the constraint that it should be the smoothest (in a given sense) while satisfying the data. As the measure of smoothness we choose the square of the velocity field gradient:

$$\left(\frac{\partial u}{\partial x}\right)^2 + \left(\frac{\partial u}{\partial y}\right)^2 + \left(\frac{\partial v}{\partial x}\right)^2 + \left(\frac{\partial v}{\partial y}\right)^2 \quad (3)$$

A variational functional provides the most



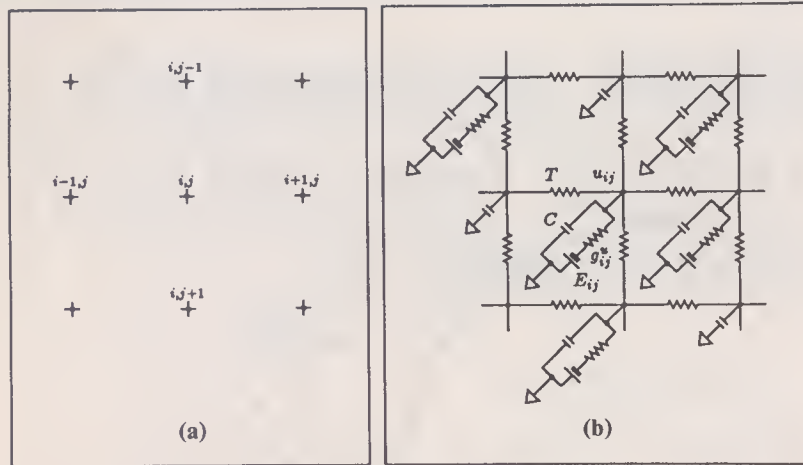


Figure 2. Rectangular grid and resistive network. (a) Rectangular grid for solving the discrete version of Equation 4. (b) Part of the resistive network minimizing the discrete approximation of the energy function E in Equation 4. We assume the conductance T connecting neighboring nodes to be constant. Each node connects to a variable battery E_{ij} via a conductance g_{ij}^u . Parasitic capacities (on the order of 0.1 picofarad) give the circuit its dynamic behavior. The final network consists of two such resistive networks superimposed, where corresponding nodes are connected via a variable conductance T_{c-ij} , as in Figure 3b. Once the batteries E_{ij} and conductances g_{ij}^u and g_{ij}^v have been set, the network will converge—following Kirchhoff's laws—to the state of least power dissipation that corresponds to the solution of the variational Equation 4.

general way to formulate the problem.² The final velocity field (u, v) minimizes

$$E(u, v) = \iint (I_x u + I_y v + I_t)^2 + \lambda \left[\left(\frac{\partial u}{\partial x} \right)^2 + \left(\frac{\partial u}{\partial y} \right)^2 + \left(\frac{\partial v}{\partial x} \right)^2 + \left(\frac{\partial v}{\partial y} \right)^2 \right] dx dy \quad (4)$$

where the regularization parameter λ is inversely dependent on the signal-to-noise ratio. The first term describes the fact that the final solution should follow as closely as possible the measured data, whereas the second term imposes the smoothness constraint on the solution. The degree of minimization of one or the other term is governed by λ . With accurate data, violating the first term should be "expensive" and λ will be small. Conversely, with unreliable data (low signal-to-noise ratio), much more emphasis will be placed on the smoothness term. Horn and Schunck² first formulated this variational approach to the motion problem.

The energy $E(u, v)$ is quadratic in the

unknown u and v . It then follows from standard calculus of variation that the associated Euler-Lagrange equations will be linear in u and v :

$$\begin{aligned} I_x^2 u + I_x I_y v - \lambda \nabla^2 u + I_x I_t &= 0 \\ I_x I_y u + I_y^2 v - \lambda \nabla^2 v + I_y I_t &= 0 \end{aligned} \quad (5)$$

We now have two linear equations at every point. Our problem is therefore completely determined. We could now use a number of iterative techniques, such as steepest descent, to solve these equations. Instead, we pursue a different path.

Analog resistive networks. Let us assume that we are formulating Equations 4 and 5 on a discrete two-dimensional grid, such as the one shown in Figure 2a. Equation 5 then transforms into

$$\begin{aligned} I_{xij}^2 u_{ij} + I_{xij} I_{yij} v_{ij} - \lambda (u_{i+1j} + u_{ij+1} \\ - 4u_{ij} + u_{i-1j} + u_{ij-1}) \\ + I_{xij} I_{tij} &= 0 \end{aligned}$$

$$\begin{aligned} I_{xij} I_{yij} u_{ij} + I_{yij}^2 v_{ij} - \lambda (v_{i+1j} + v_{ij+1} \\ - 4v_{ij} + v_{i-1j} + v_{ij-1}) \\ + I_{yij} I_{tij} &= 0 \end{aligned} \quad (6)$$

where we replaced the Laplacian with its five-point approximation on a rectangular grid. We now show that this set of linear equations can be solved naturally using a simple resistive network. Let us apply Kirchhoff's current law to the center node of the resistive network shown in Figure 2b. We then have the following update equation:

$$\begin{aligned} C \frac{du_{ij}}{dt} &= T(u_{i+1j} + u_{ij+1} - 4u_{ij} + \\ u_{i-1j} + u_{ij-1}) &+ g_{ij}^u (E_{ij} - u_{ij}) \end{aligned} \quad (7)$$

Let us now assume that we have two such resistive networks superimposed, with the node ij in the upmost network connected—via a conductance T_{c-ij} —to the appropriate node ij in the bottom network (see Figure 3b). We then have two equations similar to Equation 7 with a coupling term $T_{c-ij}(v_{ij} - u_{ij})$, where v_{ij} is the voltage at node ij in the bottom network. If we assume that the resistive network has converged to its final state, $du_{ij}/dt = 0$ and $dv_{ij}/dt = 0$, we see that both equations are identical with Equation 6, if we identify

$$\begin{aligned} T &\rightarrow \lambda \\ T_{c-ij} &\rightarrow -I_{xij} I_{yij} \\ g_{ij}^u &\rightarrow I_{xij} (I_{xij} + I_{yij}) \\ g_{ij}^v &\rightarrow I_{yij} (I_{xij} + I_{yij}) \\ E_{ij} &\rightarrow \frac{-I_t}{I_{xij} + I_{yij}} \end{aligned} \quad (8)$$

Once we set the batteries and the conductances to the values indicated in Equation 8, the network will settle—following Kirchhoff's laws—into the state of least power dissipation. The associated stationary voltages correspond to the solution sought: u_{ij} is equivalent to the x component and v_{ij} to the y component of the optical flow field. A unique and stable solution always exists, even if some of the conductances have negative values. In fact, many of the conductances connecting the lower and the upper networks (T_{c-ij}) and the conductances associated with the batteries (g_{ij}^u and g_{ij}^v) will be negative, because the sign of I_x and I_y can be either negative or positive. As we will see, this poses no serious problems, given the technology we have chosen to build resistances. (See the sidebar "Parallel computer implementation.")

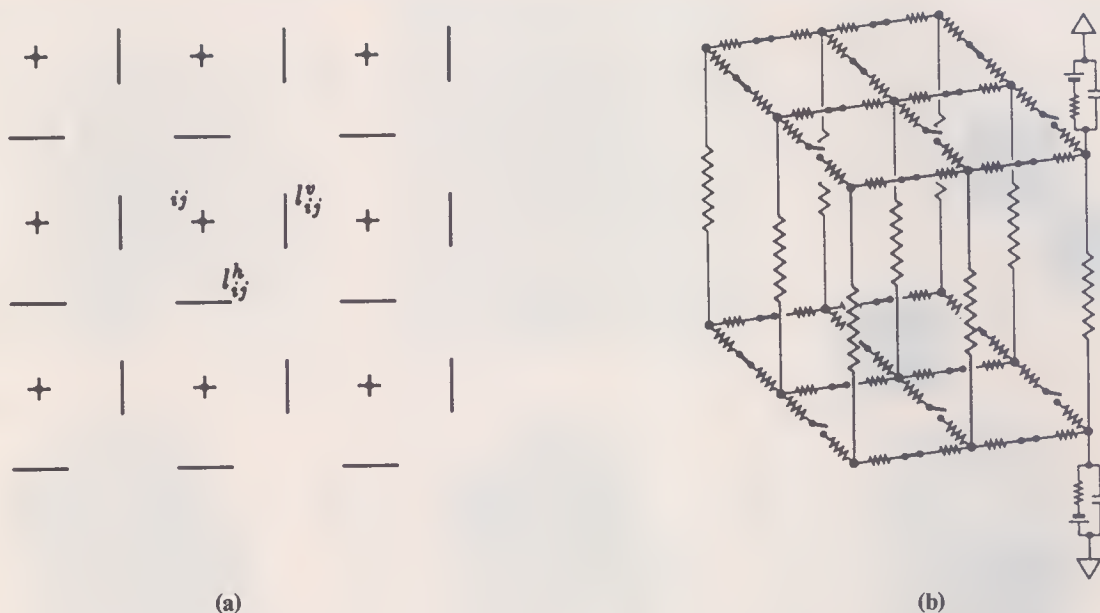


Figure 3. Rectangular grid with line processes and hybrid network. (a) The location of the horizontal (l_{ij}^h) and vertical (l_{ij}^v) line processes relative to the rectangular motion-field grid. **(b)** The hybrid resistive network, computing the optical flow in the presence of discontinuities. The conductances T_{c-ij} , which connect both grids, depend on the brightness gradient, as do the conductances g_{ij}^u and g_{ij}^v , which connect each node with the battery. For clarity, only two such elements are shown. The battery E_{ij} depends on both the temporal and the spatial gradient and is zero if no brightness change occurs. The x component of the velocity, u , is given by the voltage in the top network, while the y component of the velocity, v , is given by the voltage in the bottom network. A high voltage value at location i,j will spread to its four neighboring nodes. The degree to which voltage spreads depends on the value of the fixed conductance, T , given by the inverse of the signal-to-noise ratio. Binary switches, which make or break the resistive connections between nodes, implement motion discontinuities, because an arbitrary high voltage (velocity) will not affect the neighboring site across the discontinuity. An extended horizontal motion discontinuity is indicated. These switches could be under the control of distributed digital processors. Analog CMOS implementations of the line processes also are feasible.⁴

The sequences in Figures 4, 5, and 6 illustrate the resulting optical flow for synthetic and natural images. Figure 4c illustrates the initial velocity data and the velocity component perpendicular to the image gradient. Figure 4d shows the resulting smooth optical flow. As discussed by Horn and Schunck,² the smoothness constraint leads to a qualitatively correct estimate of the velocity field. Thus, one undifferentiated blob appears to move to the lower right and one blob to the upper left. However, at the occluding edge where both squares overlap, the smoothness assumption results in a spatial average of the two opposing velocities, and the estimated velocity is very small or zero.

Parallel computer implementation

We simulated the behavior of these networks for both synthetic and natural images by solving the previous circuit equations at each node. As boundary conditions, we copied the initial velocity data at the edge of the image into the nodes lying directly adjacent to but outside the image (zero normal derivative). We estimated the spatial and temporal derivatives l_x , l_y , and l_t using a discrete eight-point approximation.

Given the high computational cost associated with solving these elliptical equations, we used parallel com-

puters of the Hypercube family: the 32-node Mark III Hypercube at the Jet Propulsion Laboratory and a 4- and a 16-node Ncube in the laboratory at Caltech. Even though we used a variable time-step algorithm, convergence times were slow (10 minutes for a 128×128 image). Solving Equation 4 is similar to solving Poisson's equation. Thus, the number of iterations required to converge is proportional to n^2 (on an $n \times n$ -pixel image). A multigrid approach will greatly speed up the performance.

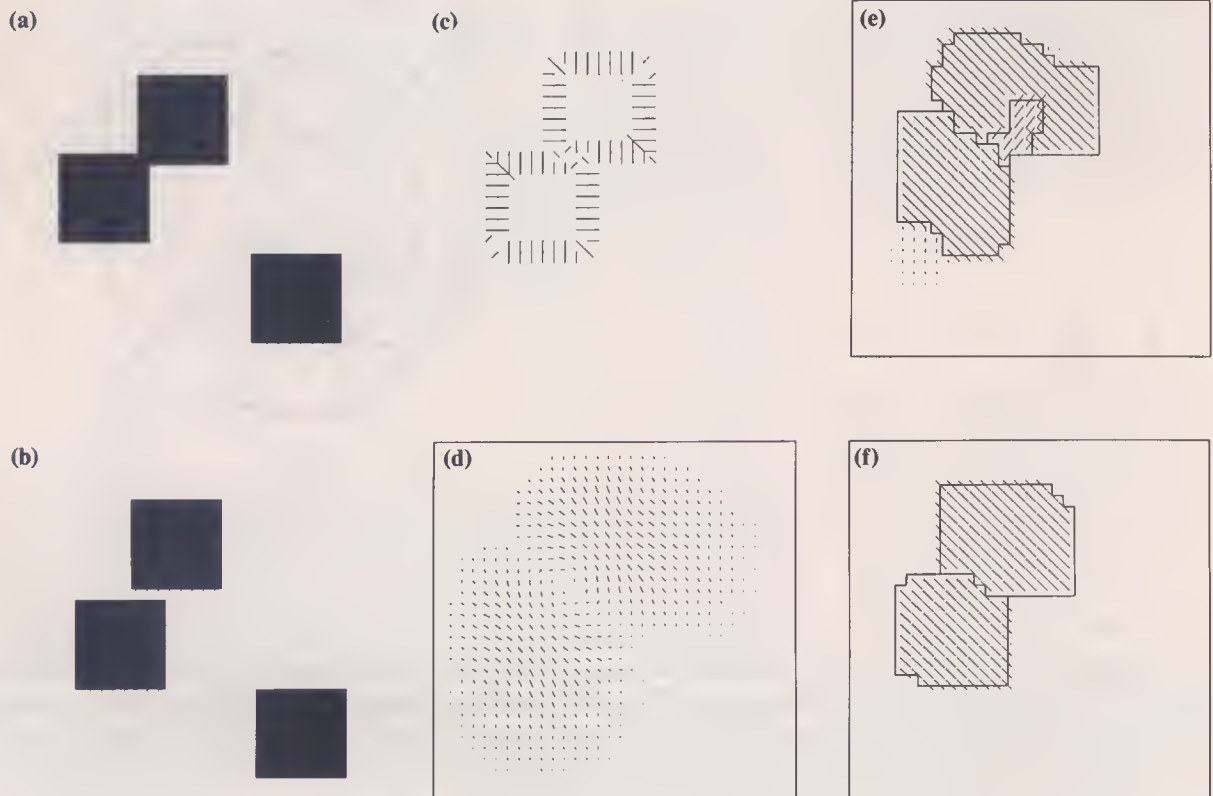


Figure 4. Motion sequence using synthetic data. (a) and (b) Two 32×32 -pixel images of three high-contrast squares on a homogeneous white background. Only the two squares on the upper left are displaced. (c) The initial velocity data. The insides of both squares contain no data. (d) The final state of the network after 240 iterations, corresponding to the smooth optical flow field. The algorithm gives a qualitatively correct estimate of the velocity field. Note, however, the vanishing velocity estimate at the occluding edges where the two moving squares overlap, caused by the averaging property of the smoothness constraint. Moreover, the moving objects are not delineated in the flow field, because the algorithm smooths over the figure-ground motion discontinuity. (e) Optical flow in the presence of motion discontinuities (indicated by solid lines). Numerous line processes are turned on in the area where the moving objects overlap. The formation of discontinuities along continuous contours is explicitly encouraged. (f) Discontinuities are strongly encouraged to form at the location of intensity edges.⁵ This additional constraint leads to the correct velocity field. The location of these discontinuities facilitates object segmentation at a later stage of visual analysis. Both (e) and (f) show the state of the hybrid network after six analog-digital cycles.

In parts of the image where the brightness gradient is zero and thus no initial velocity data exist (for instance, in the interiors of the two squares), the velocity estimate is simply the spatial average of the neighboring velocity estimates. These empty areas eventually will fill in from the boundary, similar to the flow of heat for a uniform flat plate with "hot" boundaries.

The sequence in Figure 5 also illustrates

the effect of varying the conductance T between neighboring points. As we place more confidence in the measured data (small λ), the coupling between neighboring nodes decreases because T decreases, and the optical flow becomes more inhomogeneous, better reflecting the correct velocity field. As the data becomes less reliable (large λ), more smoothing occurs until little spatial variation exists (see Figure 3).

Motion discontinuities. The smoothness assumption of Horn and Schunck² regularizes the aperture problem and leads to the qualitatively correct velocity field inside moving objects. However, this approach fails to detect the locations at which the velocity changes abruptly or discontinuously. Thus, this strategy smooths over the figure-ground discontinuity or completely fails to detect the boundary between two objects with differing veloc-

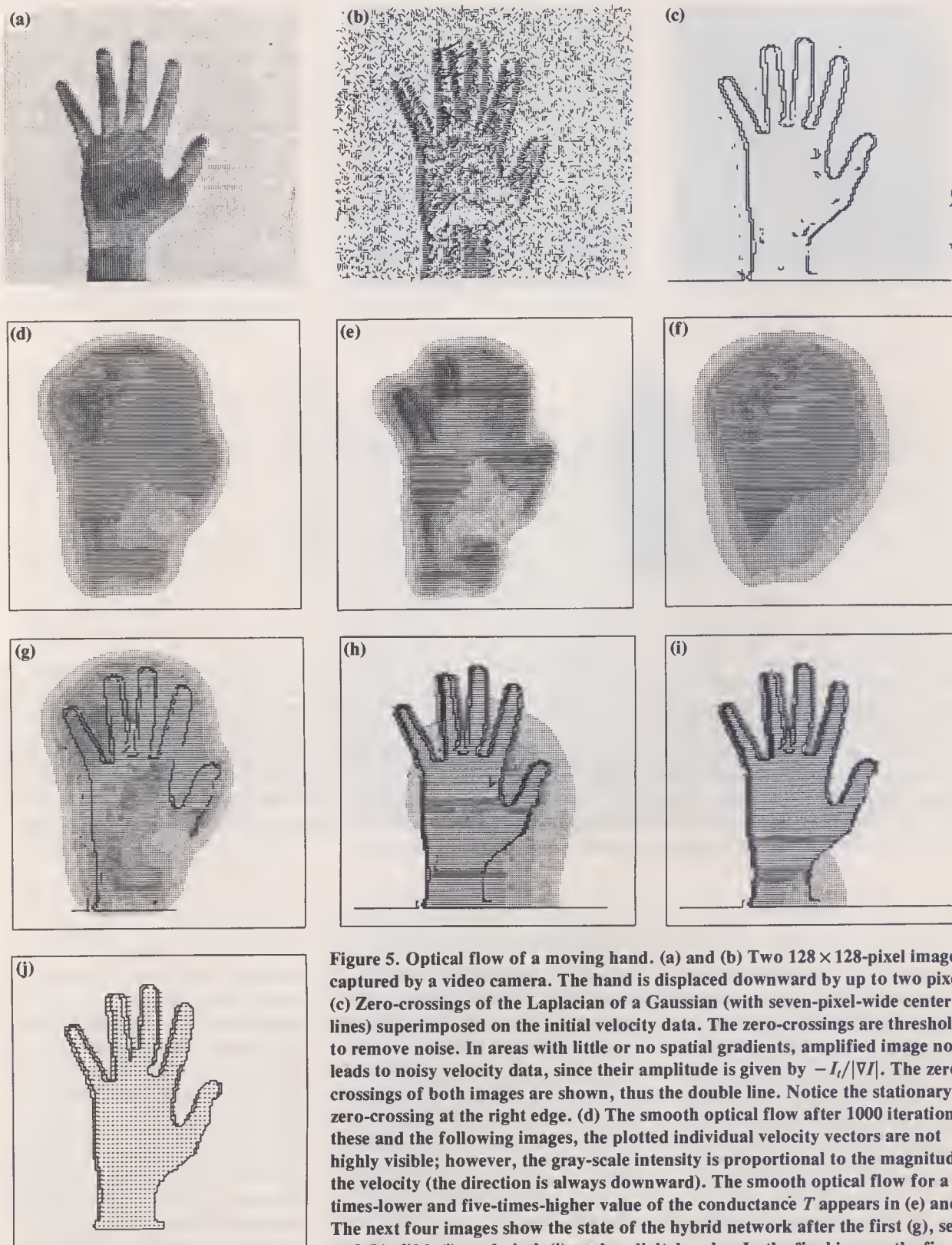


Figure 5. Optical flow of a moving hand. (a) and (b) Two 128×128 -pixel images captured by a video camera. The hand is displaced downward by up to two pixels. (c) Zero-crossings of the Laplacian of a Gaussian (with seven-pixel-wide center lines) superimposed on the initial velocity data. The zero-crossings are thresholded to remove noise. In areas with little or no spatial gradients, amplified image noise leads to noisy velocity data, since their amplitude is given by $-I_t/|\nabla I|$. The zero-crossings of both images are shown, thus the double line. Notice the stationary zero-crossing at the right edge. (d) The smooth optical flow after 1000 iterations. In these and the following images, the plotted individual velocity vectors are not highly visible; however, the gray-scale intensity is proportional to the magnitude of the velocity (the direction is always downward). The smooth optical flow for a five-times-lower and five-times-higher value of the conductance T appears in (e) and (f). The next four images show the state of the hybrid network after the first (g), second (h), fifth (i), and ninth (j) analog-digital cycles. In the final image, the fingers have a higher velocity than does the hand itself. It takes several cycles for the motion discontinuities to "creep" around the outline of the hand.

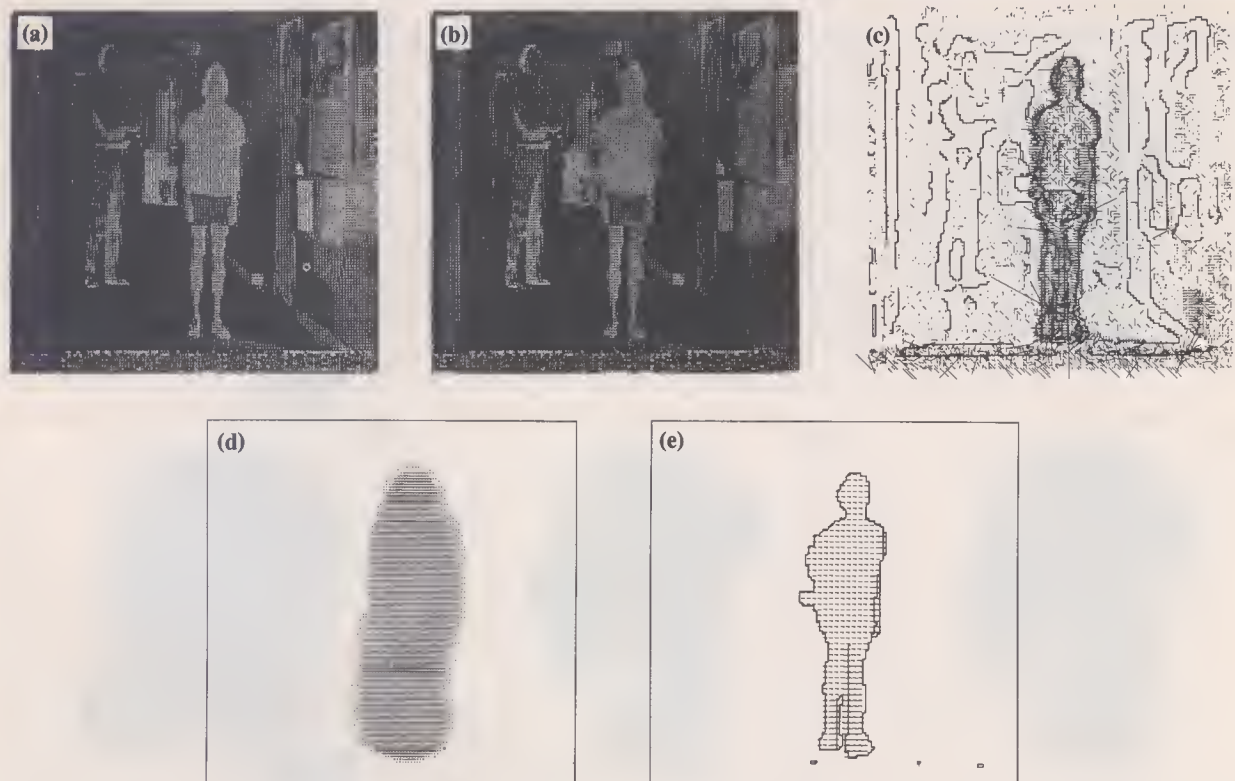


Figure 6. Optical flow of a moving person. (a) and (b) Two 128×128 pixel images captured by a video camera. The person in the foreground is moving toward the right while the person in the background is stationary. The noise in the lower part of the image is a camera artifact. (c) Thresholded zero-crossings superimposed on the initial velocity data. (d) The smooth optical flow after 1000 iterations. Note that the noise in the lower part of both video images is completely smoothed away. (e) The final piecewise smooth optical flow after 13 analog-digital cycles. The velocity field is subsampled to improve visibility. With the exception of a square appendage at the right hip, the optical flow field shown corresponds to the correct velocity field. The appendage, caused by the edge-detection scheme lumping part of the garbage can in the background with the contour of the person, represents an instance of what psychophysicists term motion capture. More recently, we have successfully computed the optical flow field for images with many, partially occluding, moving people.

ities, because the algorithm combines velocity information across motion boundaries. We argue that motion discontinuities are the most interesting locations in any image, because they indicate where one object ends and another one begins. Motion as well as intensity discontinuities are vital for solving the critical object segmentation problem that occurs at a subsequent stage of the image understanding process.

Various researchers have attempted to prevent the smoothing constraint from taking effect across strong velocity gradients.⁶ Geman and Geman⁷ proposed a successful strategy for dealing with discon-

tinuities. They exploited an analogy between statistical mechanics and images, whereby the intensity values at each picture element and the presence of discontinuities are viewed as states of particles on a lattice. We can assign an "energy" function to this system and compute its most likely state.

In this article, we do not rigorously develop this approach, based on Bayesian estimation theory.^{7,8} Suffice it to say that a priori knowledge (for instance, that the velocity field in general should be smooth) can be formulated in terms of a Markov random field model of the image. (In a Markov random field, the conditional

probability that a given variable at location i, j has a particular value f_{ij} depends only on the values of f in a neighborhood of i, j .) Given such an image model, and given noisy data, we then estimate the "best" flow field by some likelihood criterion. The one we use here is the maximum a posteriori estimate, although other possible criteria have certain advantages.⁸ Maximizing the a posteriori probability yields the best solution. We can show this to be fully equivalent to minimizing an expression such as Equation 4.

To reconstruct images consisting of piecewise constant segments, Geman and Geman⁷ further introduced the powerful

idea of a line process l (see also Blake and Zisserman⁹). For our purposes, we assume that a line process can occupy one of two states: "on" ($l=1$) or "off" ($l=0$). Line discontinuities are located on a regular lattice set between the original pixel lattice (see Figure 3a), such that each pixel ij has one horizontal l_{ij}^h and one vertical l_{ij}^v line process associated with it. If the appropriate line process is turned on, the smoothness term between the two adjacent pixels will be set to zero.

To prevent line processes from forming everywhere and to incorporate additional knowledge regarding discontinuities into the line processes, we must include an additional term $V_c(l)$ in the new energy function:

$$E(u, v, l^h, l^v) = \sum_{ij} (I_x u_{ij} + I_y v_{ij} + I_l)^2 + \lambda \sum_{ij} (1 - l_{ij}^h) [(u_{i+1j} - u_{ij})^2 + (v_{i+1j} - v_{ij})^2] + \lambda \sum_{ij} (1 - l_{ij}^v) [(u_{ij+1} - u_{ij})^2 + (v_{ij+1} - v_{ij})^2] + V_c(l) \quad (9)$$

V_c contains a number of terms penalizing or encouraging specific configurations of line processes:

$$V_c(l) = C_c \sum_{ij} l_{ij}^h + C_p \sum_{ij} l_{ij}^h (l_{ij+1}^h + l_{ij+2}^h) + C_l V(l) \quad (10)$$

plus the corresponding expression for the vertical line process l_{ij}^v (obtained by interchanging i with j and l_{ij}^h with l_{ij}^v). The first term (C_c) penalizes each introduction of a line process, because the cost C_c has to be "paid" every time a line process is turned on. The second term prevents the formation of parallel lines. If either l_{ij+1}^h or l_{ij+2}^h is turned on, this term will tend to prevent l_{ij}^h from turning on. The third term (C_l) embodies the fact that, in general, motion discontinuities occur along extended contours and rarely intersect. We adopt the function given by Koch et al.¹⁰ favoring the formation of motion discontinuities along extended contours and penalizing both multiple line intersections and isolated discontinuities.

We obtain the optical flow by minimizing the cost function in Equation 9 with respect to both the velocity field $v = (u, v)$ and the line processes l^h and l^v . However, unlike before, this cost or energy function is nonconvex, since it contains cubic and possibly higher terms (in V_c). Geman and Geman resorted to annealing, a statistical optimization technique, to find the ground state of their system. If annealing is applied appropriately, the system con-

verges with probability converging to one to the global maximum.⁷ However, the length of the required convergence times makes any practical application expensive.

To find an optimal solution to this non-quadratic minimization problem, we follow the approach used by Koch et al.¹⁰ and use a purely deterministic algorithm, based on solving Kirchhoff's equations for a mixed analog and digital network.^{8,11} Our algorithm exploits the fact that for a fixed distribution of line processes, the energy function of Equation 9 is quadratic. Thus, we first initialize the analog resistive network (see Figure 3b) according to Equation 8 and with no line processes on. The network then converges to the smoothest solution. Subsequently, we update the line processes by deciding at

each site of the line process lattice whether the overall energy can be lowered by setting or breaking the line process. We always accept the state of the line process corresponding to the lower energy configuration: l_{ij}^h will be turned on if $E(u, v, l_{ij}^h = 1, l^v) < E(u, v, l_{ij}^h = 0, l^v)$; otherwise, $l_{ij}^h = 0$. This computation requires only local information. Breaking the appropriate resistive connection between the two neighboring nodes switches on the line processes. After the completion of one such analog-digital cycle, we reiterate and compute the smoothest state of the analog network for the newly updated distribution of line processes.

Although we have no guarantee that the system will converge to the global minimum, given our use of a gradient descent

Restricting motion discontinuities to edges

As edges we use the zero-crossings of a Laplacian of a Gaussian convolved with the original image. Marr and Hildreth¹² have shown that these locations usually correspond to physical edges. We threshold these zero-crossings (based on the square of the gradient) in order to remove spurious zero-crossings caused by noise and "weak" edges. Other edge detection algorithms should work equally well.

We now add a new term $V_{z-c_{ij}}$ to our energy function E , such that $V_{z-c_{ij}}$ is zero if l_{ij} is off or if l_{ij} is on

and a zero-crossing exists between locations i and j . If $l_{ij} = 1$ in the absence of a zero-crossing, $V_{z-c_{ij}}$ is set to a large positive number (in our case, 1000).

This strategy effectively prevents motion discontinuities from forming at locations where no zero-crossings exist, unless strongly suggested by the data. Conversely, however, zero-crossings by themselves will not induce the formation of discontinuities in the absence of motion gradients.

Varying the 'amplitude' of motion discontinuities

When dealing with real data, the amplitude of velocity and, consequently, the amplitude of any motion discontinuity vary over a considerable range (as compared to the artificial situation in Figure 4). Our strategy in dealing with this problem involves varying the magnitude of the V_c term in Equation 9 by multiplying V_c with $1/K(t)$.¹⁰ Initially, $K(t)$ is small, but it then increases linearly until a given upper bound.

In other words, the formation of discontinuities is penalized initially,

encouraging a smooth interpolation everywhere except at very steep velocity gradients. Subsequently, by paying a smaller price for the formation of line processes, the optical flow will break at smaller velocity gradients. The final state of the network is independent of the speed at which $K(t)$ changes (adiabatic convergence). All other parameters remain constant and are identical for all simulations reported in this article.

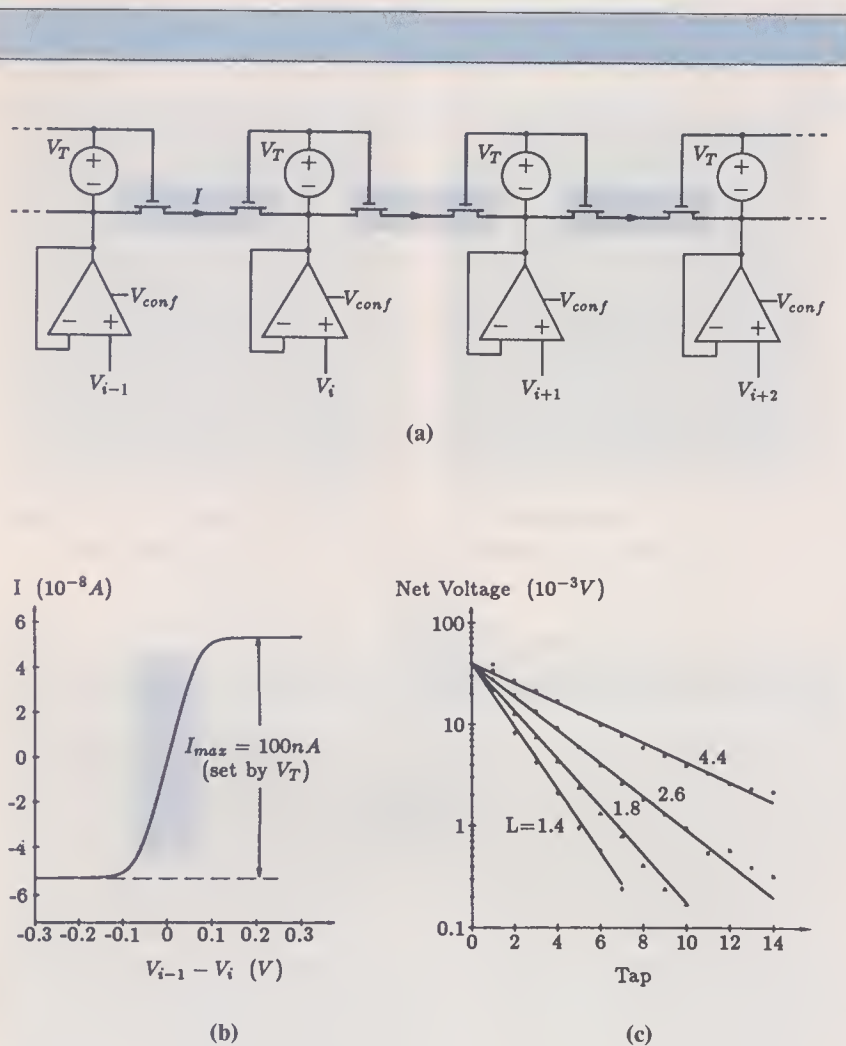


Figure 7. Basic resistance element in analog subthreshold CMOS technology. (a) Shows a one-dimensional cut through the resistive network of Figures 2 and 8. The active circuits—built out of nine transistors—within the shaded areas implement a variable nonlinear resistance. Each transconductance amplifier implements the conductance G (see Figure 8), whose value can be set by V_{conf} . The spatial response of such a network to a point voltage stimulus applied to the left-hand side is shown in (c). In an ideal tap line, the measured voltage values (points) should follow an exponential decay (lines). $V_T - V_{conf}$ sets the decay length L . The current-voltage characteristic of one such resistive element is illustrated in (b). The voltage V_T controls the maximum current and thus the slope of the resistance, which can vary between 100 k Ω and 10 G Ω . Many variations are possible.⁴

rule, the system seems to find next-to-optimal solutions (see Figures 4, 5, and 6) in about 10 to 15 analog-digital cycles. Furthermore, the algorithm must converge, because at each step the energy E is always reduced and E is bound from below. We compared statistical annealing with our deterministic method in the case of interpolating and smoothing sparsely

sampled data in the presence of discontinuity, where the underlying energy function is similar to E in Equation 9. Both methods converged to qualitatively similar solutions.¹⁰

The synthetic motion sequence in Figure 4 demonstrates the dramatic effect of the line processes. The optical flow outside the discontinuities approximately delineating

the boundaries of the moving squares is zero, as it should be (see Figure 4e). Where the two squares overlap, however, the velocity gradient is high and multiple intersecting discontinuities exist.

To restrict further the location of discontinuities, we adopt a technique used by Gamble and Poggio⁵ to locate depth discontinuities by requiring that depth discontinuities coincide with the location of intensity edges. In general, the physical processes and the geometry of the three-dimensional scene giving rise to the motion discontinuity will also give rise to an intensity edge. For example, moving physical objects occluding other objects will give rise to an image with edges at the occluding boundaries. In fact, only under laboratory conditions—for instance, using random dot patterns—does a motion discontinuity not coincide with intensity edges.

Figure 4f demonstrates that this strategy leads to the correct velocity field—with the exception of the corners—in addition to labeling all motion discontinuities. Figures 5 and 6 demonstrate our method on image pairs obtained with a video camera. See also the sidebars “Restricting motion discontinuities to edges” and “Varying the ‘amplitude’ of motion discontinuities.”

Analog VLSI networks

Even with the approximations and optimizations we previously described, the computations involved in this and similar early vision tasks require tens of minutes to hours on a large-scale parallel computer (see, however, Gamble and Poggio⁵). For the computations to be truly useful, we should be able to carry them out on a whole image in real time. Fortunately, modern integrated circuit technology gives us a medium in which we can realize extremely complex, analog real-time implementations of these computational metaphors.⁴

We can achieve a compact implementation of a resistive network using an ordinary CMOS process, provided the transistors run in the subthreshold range, where their characteristics are ideal for implementing low-current analog functions. We achieve the effect of a resistor by choosing the circuit configuration shown in Figure 7 rather than by using the resistance of a special layer in the process. We can control the value of the resulting resistance over five orders of magnitude by setting the bias voltages on the upper and

lower current source transistors. The current-voltage curve saturates above approximately 100 millivolts, a feature that we can use to advantage in many applications.

With small voltage gradients, we can treat the circuit as if it were a linear resistor, as shown by the shaded areas on the curves (Figure 7b). Conductances to signal input sources are implemented with transconductance amplifier followers, as shown in Figure 7a. Each amplifier injects a current into the network proportional to the difference between the local signal potential and the potential of the network. The effect of a conductance is thus achieved without drawing any current from the signal source. The value of the conductance is set by the transconductance control on the amplifier, which we can use to reflect the confidence assigned to the particular input. High conductance values give the network a short spatial-averaging scale, low values give a long averaging scale.

Figure 7c shows the spatial response of an experimental one-dimensional network to a point stimulus. We obtained the different values of averaging length L by appropriate settings of the amplifier transconductances. We can easily realize resistances with an effective negative resistance value.

Figure 8 shows the ideal configuration for a network implementation in two dimensions. Each point on the hexagonal grid is coupled to six equivalent neighbors. The high degree of symmetry of such an arrangement creates a nearly isotropic environment, free of many of the "preferred axis artifacts" introduced by an orthogonal grid. In addition, the larger connectivity allows a greater variation in effective resistor value caused by variations in transistor parameters.

Figure 9 shows a test chip implementing this network. Each node includes the resistor apparatus and a set of sample-and-hold circuits for setting the confidence and signal input voltages. In addition, an output amplifier enables measurement of the node voltage without disturbing the node itself. A scanning mechanism addresses both the sample-and-hold circuits and the output buffer, so the stored variables can be refreshed or updated, and the map of node voltages can be read out in real time.

A 48×48 silicon retina has been constructed that uses the hexagonal network of Figure 8 as a model for the horizontal cell layer in the vertebrate retina.¹³ In this application, the input potentials were the

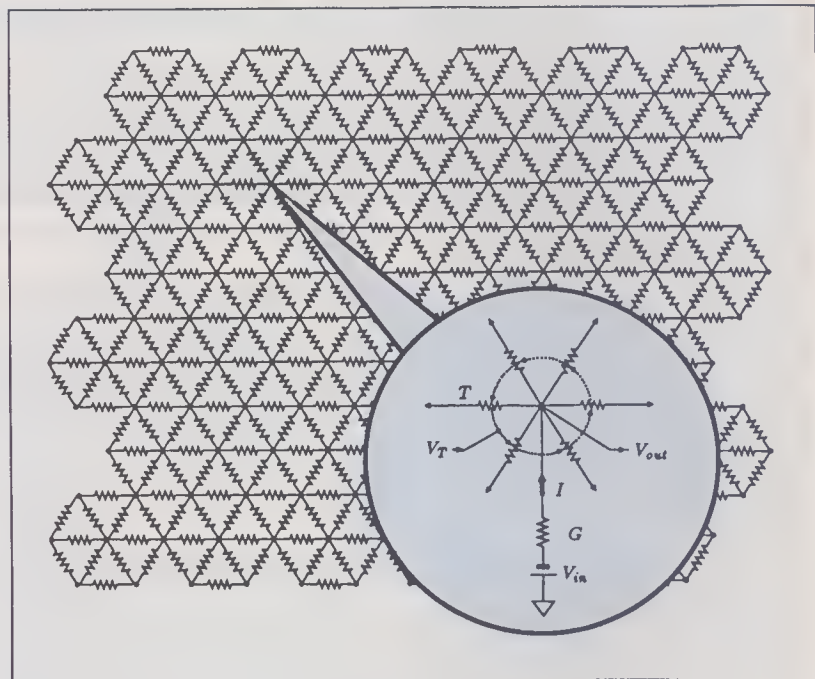


Figure 8. Circuit design for a resistive network for interpolating and smoothing noisy and sparsely sampled depth measurements.¹⁰ The basic version of this CMOS circuit contains 20×20 grid points on a hexagonal lattice. The individual resistive elements with a variable slope controlled by V_T , shown in Figure 7, correspond to the term governing the smoothness, λ . At those locations where a depth measurement d_{ij} is present, the battery is set to this value ($V_{in} = d_{ij}$) and the value of the conductance G is set to $1/(2\sigma^2)$, where σ^2 is the variance of the Gaussian noise process associated with the depth measurements. If no depth data are present at that node, G is set to zero. The voltage at each node corresponds to the discrete values of the smoothed surface fitted through the noisy and sparse measurements.^{8,10} The network for computing smooth optical flow minimizing E in Equation 4 via the network shown in Figure 3b is similar to this circuit.

outputs of logarithmic photoreceptors—implemented via phototransistors—and the potential difference across the conductance T formed an excellent approximation to the Laplacian operator.¹² This model results in the classical center-surround receptive field properties observed in the response of retinal ganglion cells. The circuit performs in real time.

We have demonstrated that the introduction of binary motion discontinuities into Horn and Schunck's² algorithm leads to a much

improved performance of their method, particularly for the optical flow in the presence of a number of moving objects. Moreover, we have shown that the appropriate computations map onto simple resistive networks.

We are now implementing these resistive networks in VLSI circuits, using sub-threshold CMOS technology. Many problems in early vision can be formulated in terms of similar nonconvex energy functions that need to be minimized, such as binocular stereo, edge detection, surface interpolation, and structure from motion.^{3,8,11} A similar approach to early

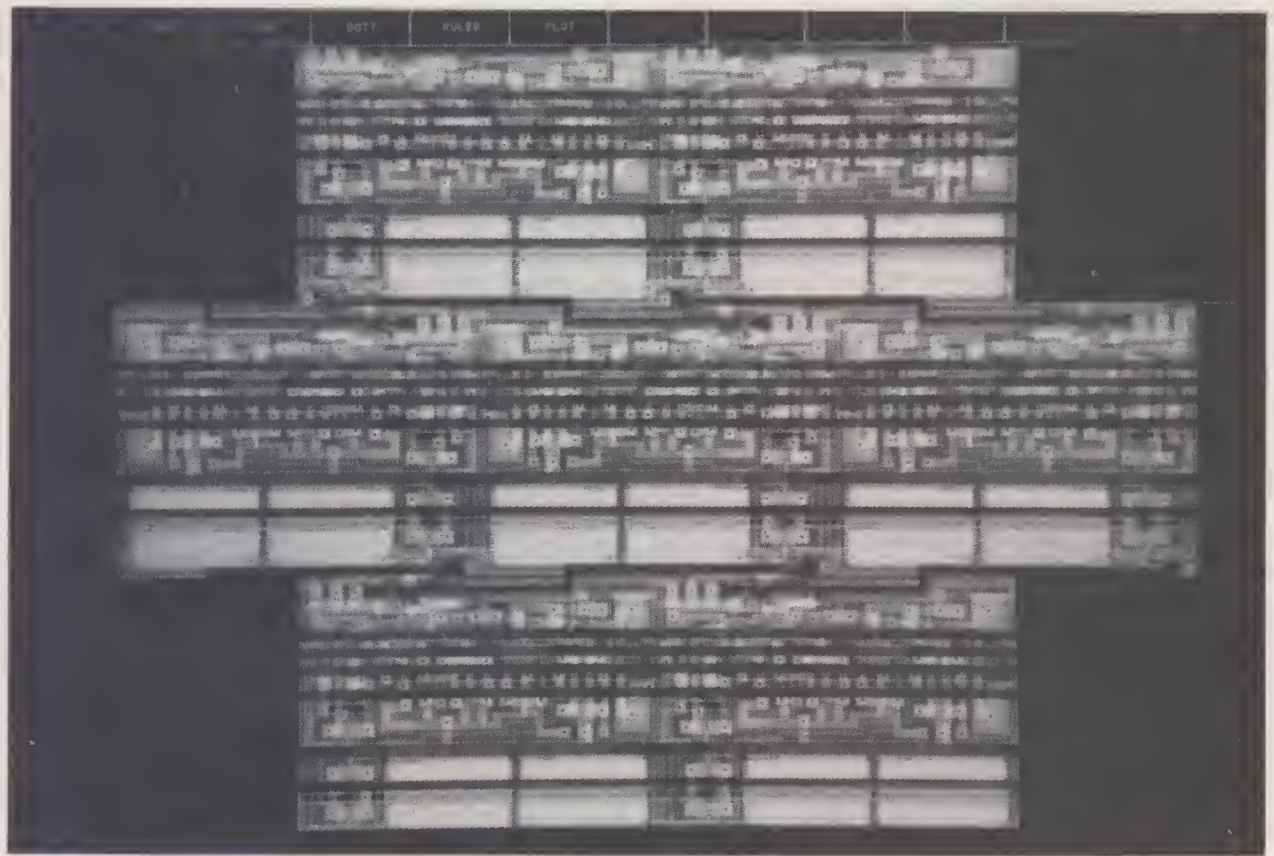


Figure 9. Partial layout of the CMOS chip implementing the resistive network shown in Figure 8, which interpolates sparsely sampled noisy data. Only seven cells (out of the 20×20 cell array) are shown in order to demonstrate the hexagonal grid. Each cell is dominated by the two capacities (approximately two picofarads each) for holding the depth data and its associated confidence value, consists of 46 transistors, and measures $180 \times 132 \lambda^2$. For a $\lambda = 1.5 \mu\text{m}$ production run, the total chip measures $5.4 \times 2.6 \text{ mm}$. If image acquisition-devices (phototransistors) are placed on the chip, the sample-and-hold circuitry can be eliminated, substantially reducing the area of the elementary cell.¹³

Fault tolerance

Hutchinson and Koch¹⁵ demonstrated the robustness of these resistive networks to component errors. In their circuit simulations of the resistive network for interpolating surfaces from noisy and sparsely sampled data (shown in Figures 2b and 8), they replaced each transversal conductance T by $T(1 + N(\sigma^2))$, where N is a zero-mean Gaussian probability distribution with variance σ^2 .

Due to the linearity of the network connections, errors "average out" and performance is only marginally

impaired even for widely varying conductances ($\sigma^2 = 0.5$). If nodes are pulled accidentally to ground, the line processes in their immediate neighborhood turn on because of the high voltage gradient, isolating these nodes and preventing error propagation. The saturation characteristic of our resistive elements outside their linear range (see Figure 7b) serves to prevent high current flows, because high voltage gradients between neighboring nodes induce only a constant maximal current.⁴

vision—using the fine-grained, mesh-type, single-instruction, multiple-data parallel Connection Machine instead of resistive networks—is being pioneered at MIT's Artificial Intelligence Laboratory in the Vision Machine project.^{5,14}

These networks share several features with biological neuronal networks. Specifically, they do not require a system-wide clock, they rely on many connections between simple computational nodes, they converge rapidly (within several time constants), and they are quite robust to hardware errors. (See the sidebar "Fault tolerance.")

Our networks consume moderate amounts of power, because each resistive

element operates in the millivolt and 10 nanoampere range. The entire retina chip¹³ requires about 100 μ W (the dominant power consumption lies in the photoconversion stage).

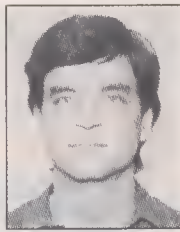
These features—real-time performance, low power consumption, robustness, and small spatial dimensions—make these circuits attractive for a variety of deep space missions. In collaboration with the Jet Propulsion Laboratory, we are currently evaluating the feasibility of such resistive network-based vision systems for autonomous vehicles to be used in the exploration of planetary surfaces, such as that of Mars. □

Acknowledgments

An early version of this model was developed and implemented in collaboration with A.L. Yuille (1987). Mathew Avalos and Andrew Hsu wrote the code for the Imaging Technology system, and Erik Staats for the Neube. Koch is supported by an Office of Naval Research Young Investigator Award, grants from the NSF Advanced Engineering Program (EET-8714710 and IST-8700064), a grant from the Dept. of Energy (DE-FGO3-85 ER25009), and the James Lee Powell Foundation. Mead is also supported by the Office of Naval Research and the System Development Foundation. A portion of this research was carried out at the Jet Propulsion Laboratory, California Institute of Technology, and was sponsored by NASA.

References

1. E.C. Hildreth, *The Measurement of Visual Motion*, MIT Press, Cambridge, Mass., 1984.
2. B.K.P. Horn and B.G. Schunck, "Determining Optical Flow," *Artificial Intelligence*, Vol. 17, 1981, pp. 185-203.
3. T. Poggio, V. Torre, and C. Koch, "Computational Vision and Regularization Theory," *Nature*, Vol. 317, 1985, pp. 314-319.
4. C. Mead, *Analog VLSI and Neural Systems*, Addison-Wesley Publishing Co., Reading, Mass., to appear in the summer of 1988.
5. E. Gamble and T. Poggio, "Integration of Intensity Edges with Stereo and Motion," *Artificial Intelligence Lab. Memo No. 970*, MIT, Cambridge, Mass., 1987.
6. H.H. Nagel, "On the Estimation of Optical Flow: Relations Between Different Approaches and Some New Results," *Artificial Intelligence*, Vol. 33, 1987, pp. 299-324.
7. S. Geman and D. Geman, "Stochastic Relaxation, Gibbs Distribution, and the Bayesian Restoration of Images," *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol. 6, 1984, pp. 721-741.
8. J. Marroquin, S. Mitter, and T. Poggio, "Probabilistic Solution of Ill-Posed Problems in Computational Vision," *J. Am. Statistics Assoc.*, Vol. 82, 1987, pp. 76-89.
9. A. Blake and A. Zisserman, *Visual Reconstruction*, MIT Press, Cambridge, Mass., 1987.
10. C. Koch, J. Marroquin, and A. Yuille, "Analog 'Neural' Networks in Early Vision," *Proc. Natl. Academy of Sciences USA* 83, 1986, pp. 4263-4267.
11. A.L. Yuille, "Energy Functions for Early Vision and Analog Networks," *Artificial Intelligence Lab. Memo No. 987*, MIT, Cambridge, Mass., 1987.
12. D. Marr and E.C. Hildreth, "Theory of Edge Detection," *Proc. Royal Society of London*, Vol. B 207, 1980, pp. 187-217.
13. M.A. Sivilotti, M.A. Mahowald, and C.A. Mead, "Real-Time Visual Computations Using Analog CMOS Processing Arrays," in *1987 Stanford VLSI Conference*, P. Losleben, ed., 1987, pp. 295-312.
14. T. Poggio and the staff, "MIT Progress in Understanding Images," in *Image Understanding Workshop*, Vol. 1, DARPA, 1987, pp. 1-14.
15. J.M. Hutchinson and C. Koch, "Simple Analog and Hybrid Networks for Surface Interpolation," in *Neural Networks for Computing*, J.S. Denker, ed., American Institute of Physics, New York, 1986, pp. 235-239.



James Hutchinson has been a member of the technical staff at Thinking Machines Corp. since December 1987. He is currently implementing a document retrieval system for the Connection Machine. Previously, while at the Jet Propulsion Laboratory in Pasadena, he investigated artificial neural network applications on the Caltech/JPL Hypercube concurrent computer. His current interests include parallel computation, intelligent systems, and machine vision.

Hutchinson received the MS and BS in computer science from MIT in 1986.



Christof Koch is an assistant professor of computation and neural systems at the California Institute of Technology. He is studying early vision algorithms, such as stereo, motion, and color, in both the mammalian visual system and in simple resistive networks for machine vision. His other major interest is the biophysics of computation—understanding the biophysical and biochemical mechanisms underlying information processing in nerve cells.

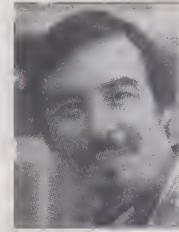
Koch studied physics and philosophy at the University of Tübingen in West Germany, receiving a PhD in 1982 at the Max Planck Institute for Biological Cybernetics.

Readers may write to Koch at Caltech, Division of Biology 216-76, Pasadena, CA 91125.



Jin Luo is a staff research assistant in both computer science and biology at the California Institute of Technology. He studied analog VLSI under Carver Mead and received an MS in computer science from Caltech in 1987.

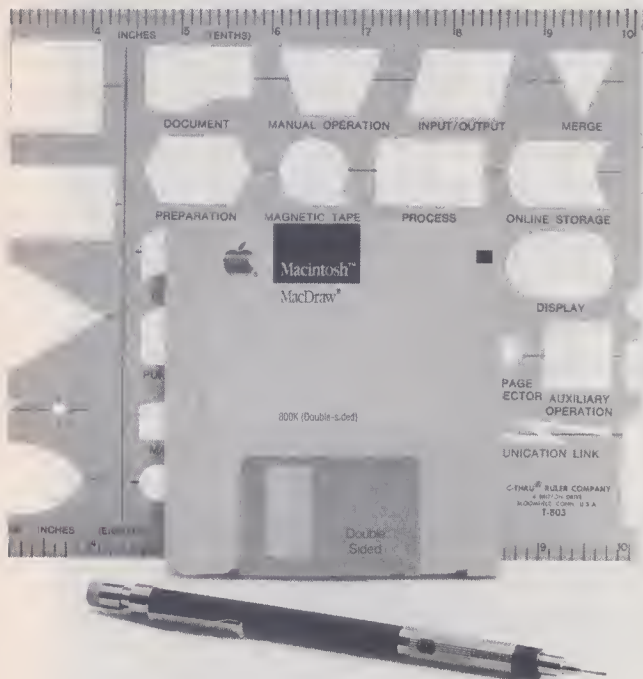
A citizen of the People's Republic of China, Luo received his BS in computer engineering from the Beijing Institute of Technology in 1983.



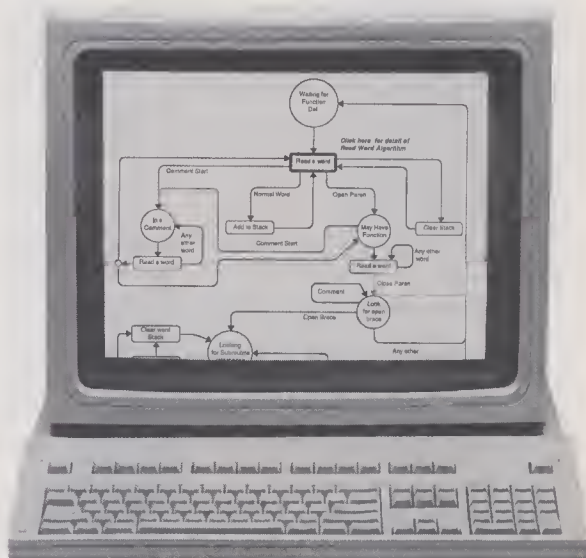
Carver Mead is the Gordon and Betty Moore Professor of Computer Science at the California Institute of Technology, where he has taught for more than 20 years. A pioneer in the fields of solid-state electronics and the management of complexity in the design of VLSI circuits, he has been heavily involved in the development of a design methodology for VLSI. His recent work concerns modeling neuronal structures, such as the retina or cochlea, using analog VLSI systems.

Mead is a fellow of the American Physical Society, a member of the National Academy of Engineering, and a foreign member of the Royal Swedish Academy of Engineering Science, and has been elected to life membership in the Franklin Institute. He has also received a number of awards.

If you just need a drawing tool, use one of these.



If you need a design tool,
use Design/2.0™



A lot of tools can help you draw. But how many can help you design and keep track of complex system models?

Meta Software introduces Design/2.0, the integrated text and graphics processor that helps you design systems, flow charts, organizational charts, computer programs, communications networks, production line process diagrams, business plans and more. Do it all faster and more efficiently than with manual tools or any other program.

Design/2.0 puts an end to time-consuming drawing and redrawing. Because it understands that objects in a diagram are related. Once you connect one object to another in a Design/2.0 diagram, it stays connected. If you move or resize an object, Design/2.0 automatically redraws all associated objects, arrows, and subordinate objects.

You can also create and edit text in your diagrams. And associate text with any object or connector. Even establish hypertext links that connect text across multiple pages.

As your model evolves, you can move detail to a subpage. In fact, you can build multi-level diagrams up to 9,999 pages. Design/2.0 automatically maintains the relationship and displays the hierarchy. So the big picture is easier to see. No matter how complex your model.

And now, trade in MacDraw or any PC drawing program for Design/2.0 by May 31, 1988 and get \$25. Just send us your Design/2.0 registration card, indicating which drawing program you use. We'll mail you a check for \$25. And start spending your time designing instead of drawing.

For more information about Design/2.0, call toll-free 1-800-227-4106.

Inside Massachusetts, 617-576-6920. Or return this coupon to: Meta Software Corporation, 150 CambridgePark Drive, Cambridge, Massachusetts 02140.

Name _____

Title

Company

Address

City

State

Zip

Telephone



Meta Software

IECO 3/88

Available for IBM® PC-AT, PS/2 & close compatibles. (Requires Windows™) \$350
Available for the Macintosh™ Plus, SE, II. \$250

IBMPC-AT and PS/2 are registered trademarks of International Business Machines Corporation.
Windows is a trademark of Microsoft Corporation. Macintosh is a trademark and
MacDraw is a registered trademark of Apple Computer, Inc.
Design/2.0 is a trademark of Meta Software Corporation.

Reader Service Number 5



A Neural Network for Visual Pattern Recognition

Kunihiko Fukushima

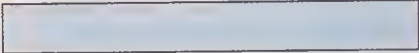
NHK Science and Technical Research Laboratories

Visual pattern recognition such as reading characters or distinguishing shapes—a task easily accomplished by human beings—presents significant difficulties to those attempting to design information processors that can do the same thing. The solution to this design dilemma apparently resides in the brain itself.

The human brain has more than 10 billion neural cells, which have complicated interconnections and constitute a large-scale network. Hence, uncovering the neural mechanisms of the higher functions of the brain is not easy. In the conventional neurophysiological approach, for example, a microelectrode is used to record the response of these cells. However, the recording can be made from, at most, a few cells simultaneously. Although we can obtain fragmentary knowledge thus, understanding the mechanism of the network as a whole proves difficult.

A modeling approach, which is a synthetic approach using neural network models, therefore continues to gain importance. In the modeling approach, we study how to interconnect neurons to synthesize a *brain model*, which is a network with the same functions and abilities as the brain.

When synthesizing a model, we try to follow physiological evidence as faithfully as possible. For parts not yet clear, however, we construct a hypothesis and synthesize a model that follows the hypothesis. We then analyze or simulate the behavior of the model and compare it



This model of the neural network offers insight into the brain's complex mechanisms as well as design principles for new information processors.

with that of the brain. If we find any discrepancy in the behavior between the model and the brain, we change the initial hypothesis and modify the model following a new hypothesis. We then test the behavior of the model again. We repeat this procedure until the model behaves in the same way as the brain. Although we must still verify the validity of the model by physiological experiment, it is probable that the brain uses the same mechanism as the model, because both respond in the same way. Hence, modeling neural networks promises to help us uncover the mechanism of the brain.

The relationship between modeling neu-

ral networks and neurophysiology resembles that between theoretical physics and experimental physics. Modeling takes a synthetic approach, while neurophysiology or psychology takes an analytical approach.

Once we complete a model, its simplification makes it easy to see the essential algorithm of information processing in the brain. We can use the algorithm directly as a design principle for new information processors.

Modeling neural networks is useful in explaining the brain and also in engineering applications. It brings the results of neurophysiological and psychological research to engineering applications in the most direct way possible.

This article discusses a neural network model thus obtained, a model with selective attention in visual pattern recognition.^{1,2}

Researchers have reported various models capable of visual pattern recognition, models that have the function of self-organization and can learn to recognize patterns. Many are hierarchical networks consisting of layers of neuron-like cells. The ability to process information increases in proportion to the number of layers in the network. Various studies have attempted to find effective learning procedures for the self-organization of multilayered networks. An example of *learning-with-a-teacher* (supervised learning) useful for training multilayered networks is the back-propagating errors

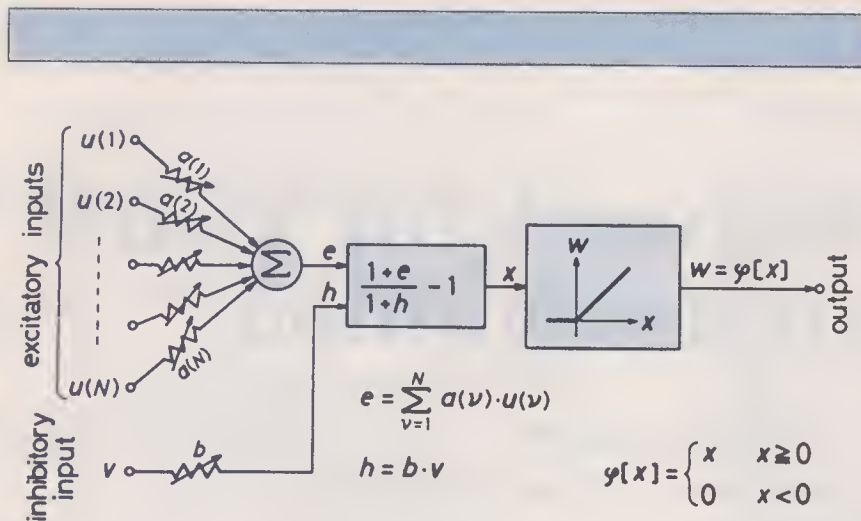


Figure 1. Input-to-output characteristics of a u_s cell: a typical example of the cells employed in the neural network model.⁷ The strength of the input connections (or weights), $a(1)$, $a(2)$, ..., and b , is variable and reinforced during the process of self-organization of the network. The excitatory effect e , which is the weighted sum of all the excitatory inputs, is suppressed by the inhibitory effect h in a shunting manner. When the inhibition is stronger than the excitation, the output of the cell becomes zero. Some of the cells in the network have fixed input connections formed from the beginning and not variable. In some kinds of cells, a nonlinear summation of excitatory inputs is performed.

procedure.³ The *cognitron*⁴ model, which I proposed in 1975, uses *learning-without-a-teacher* (unsupervised learning). The procedure used in the *cognitron* has been classified under the competitive-learning paradigm.⁵

The *cognitron*, like many other models, does not have the ability to correctly recognize position-shifted or shape-distorted patterns. The conventional *cognitron* usually recognizes the same pattern presented at a different position as a different pattern. I proposed the *neocognitron*^{6,7} to eliminate this defect. The *neocognitron* has the ability to recognize stimulus patterns correctly, even if the patterns are shifted in position or distorted in shape.

When two or more patterns are presented simultaneously, however, the *neocognitron* does not always correctly recognize them. To improve the function of the *neocognitron*, backward connections were added to the conventional *neocognitron*, which had only forward connections. The new model thus obtained acquired the function of selective attention

in visual pattern recognition. This model, discussed in this article, can automatically segment and recognize individual patterns presented simultaneously. The model can also restore imperfect patterns and eliminate noise from contaminated patterns.

Although various models of associative memory have reportedly been able to recall complete patterns from imperfect ones,⁸ most do not work well unless the stimulus pattern is identical in size, shape, and position to a training pattern. In contrast to such earlier models, the new model works well even for deformed stimulus patterns, regardless of their position.

Physiology

In the visual area of the cerebrum, neurons respond selectively to local features of a visual pattern, such as lines and edges in particular orientations.⁹ In the area higher than the visual cortex, cells exist that respond selectively to certain figures like circles, triangles, squares, or even human faces.¹⁰ Accordingly, the visual

system seems to have a hierarchical structure, in which simple features are first extracted from a stimulus pattern, then integrated into more complicated ones. In this hierarchy, a cell in a higher stage generally receives signals from a wider area of the retina and is more insensitive to the position of the stimulus.

Within the hierarchical structure of the visual system are forward (afferent, or bottom-up) and backward (efferent, or top-down) signal flows. For example, anatomical observations show that the major visual areas of the cerebrum interconnect in a precise topographical and reciprocal fashion.

Such neural networks in the brain are not always complete at birth. They develop gradually, neurons extending branches and making connections with many other neurons, adapting flexibly to circumstances after birth.

This kind of physiological evidence suggests a network structure for the model.

Outline of the model

The model is a network consisting of neuron-like cells of the *analog* type; that is, their inputs and outputs take non-negative analog values, corresponding to the instantaneous firing frequencies of biological neurons. Figure 1 shows a typical example of the cells employed in the network.

The network has a hierarchical multilayered structure consisting of a cascade of many layers of cells, as shown in Figure 2. The network has forward and backward connections between cells. In this hierarchy, the forward signals manage the function of pattern recognition, while the backward signals manage the function of selective attention, pattern segmentation, and associative recall.

Some of the connections between cells are variable, and the network can acquire the ability to recognize patterns by *learning-without-a-teacher*. The network can be trained to recognize any set of patterns. During the process of learning, variable connections grow gradually in accordance with the stimuli given the network. The repeated presentation of a set of training patterns is sufficient for the self-organization of the network; it does not need information about the categories into which these patterns should be classified.

When a composite figure consisting of two or more patterns is presented to the

model that has finished learning, the model selectively focuses its attention on one pattern after another, segments the pattern from the others, and recognizes it separately. Even if noise or defects affect the pattern, the model can recognize it and recall the complete pattern in which the noise has been eliminated and the defects corrected. Perfect recall does not require that the stimulus pattern be identical in shape to the training pattern the model learned. A pattern distorted in shape or changed in size can be correctly recognized and the missing portions restored.

Figure 3 schematically illustrates the signal flow in the network. In this diagram, layers of cells in the forward paths and the backward paths are drawn separately.

A stimulus pattern is presented to the lowest stage of the forward paths, the *input layer*, which consists of a two-dimensional array of receptor cells. The highest stage of the forward paths is the *recognition layer*. After the process of learning ends, the final result of the pattern recognition shows in the response of the cells of the highest stage. In other words, cells of the recognition layer work as *gnostic cells* (or *grandmother cells*); usually one cell is activated, corresponding to the category of the specific stimulus pattern. Pattern recognition by the network occurs on the basis of similarity in shape among patterns, unaffected by deformation, changes in size, and shifts in the position of the input patterns.

The output of the recognition layer is sent to lower stages through the backward paths. The forward and the backward signals interact with each other in the hierarchical network. The backward signals facilitate the forward signals and, at the same time, the forward signals gate the backward signal flow.*

The result of associative recall appears in the lowest stage of the backward paths, the *recall layer*. We can also interpret the output of the recall layer as the result of segmentation. The response of the recall layer is fed back to the input layer.

At each stage of the hierarchical network, several kinds of cells exist. Notation such as u_s , u_c , w_s , and x_c denotes cells, where letters u and w indicate the cells in the forward paths and backward paths, respectively. Figure 4 illustrates how the different cells, represented by circles, are connected to each other. Although the fig-

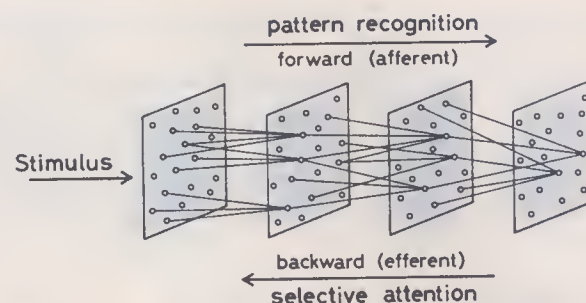


Figure 2. Hierarchical network structure and the signal flow in the network.

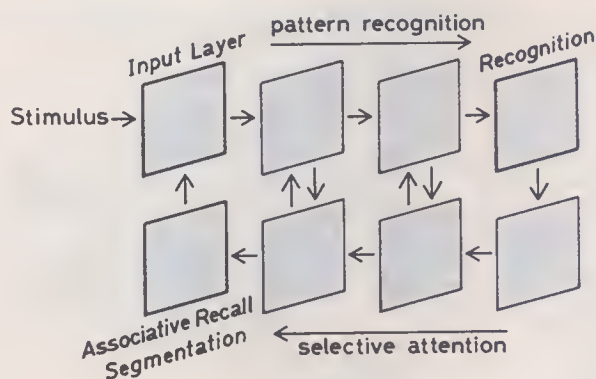


Figure 3. The forward and backward signals and their interaction in the hierarchical network.

ure shows only one of each kind of cell in each stage, numerous cells actually exist arranged in a two-dimensional array. For example, the notation u_{cl} denotes a u_c cell in the l -th stage of the hierarchical network. The L -th stage is the highest stage. In Figure 4, $L=3$. The notation U_{cl} denotes the layer of u_{cl} cells. In Figure 4, layer U_{c0} is the input layer, layer U_{c3} is the recognition layer, and layer W_{c0} is the recall layer.

Between the cells in Figure 4 are connections denoted by single lines or by double lines. A single line indicates one-to-one connections between the two groups of

cells; a double line indicates converging or diverging connections between them.

A more detailed illustration of the spatial interconnections between neighboring cells appears in Figure 5. The functions of these cells will be considered next.

Forward paths in the network

If we consider the forward paths in the network only, the model has almost the same structure and function as the neocognitron^{6,7} neural network model. In

*This process resembles that of the adaptive resonance theory¹¹ in the sense that the forward and backward signals interact with each other, but the method of interaction differs.

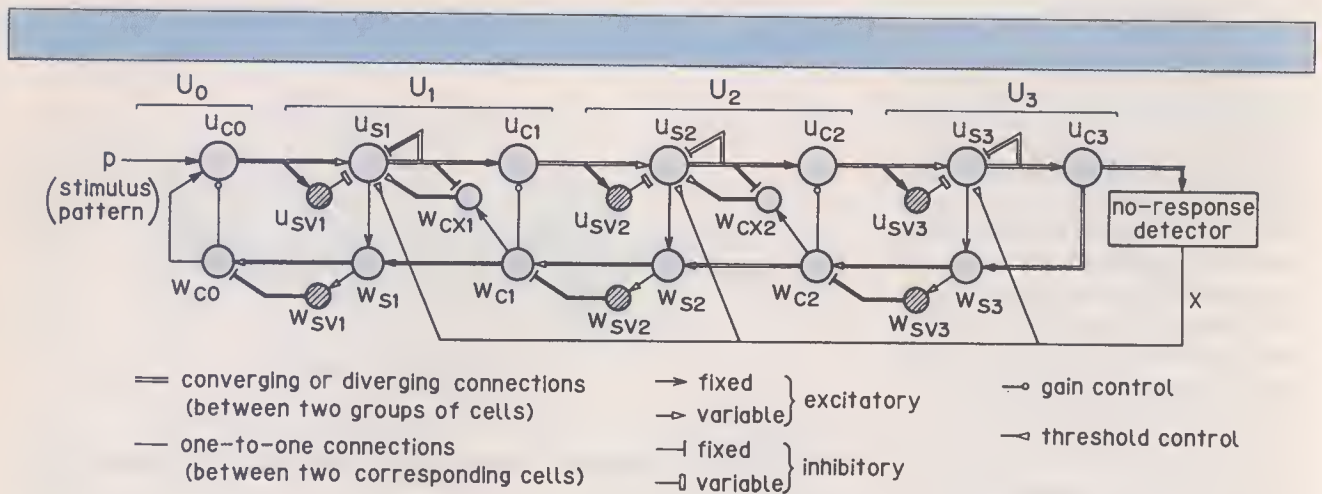


Figure 4. Hierarchical structure of the interconnections between different kinds of cells.²

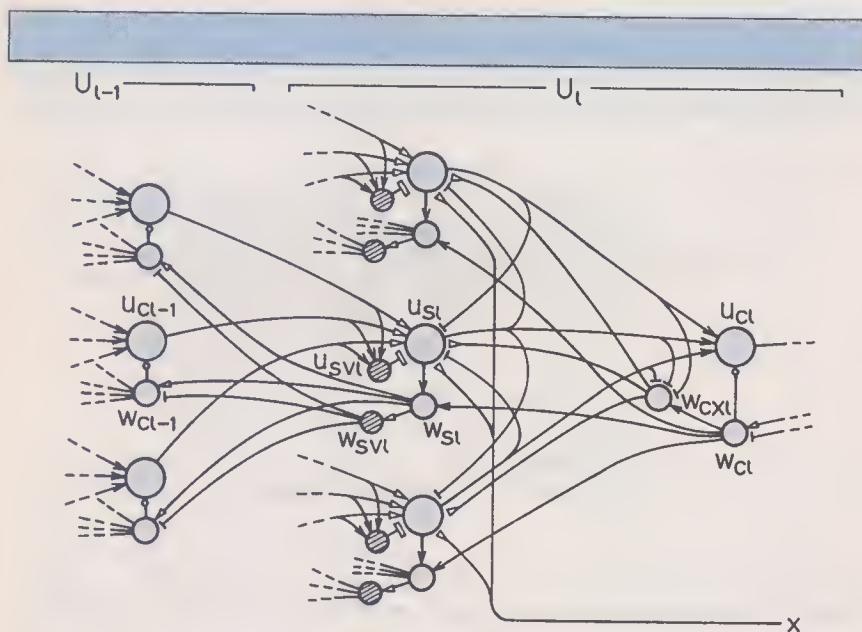


Figure 5. Detailed diagram illustrating spatial interconnections between neighboring cells.

the forward paths of the network, layers of u_S cells and u_C cells are arranged alternately, as shown in Figures 4 and 5.

Cells denoted by u_S are feature-extracting cells. Connections converging to these cells are variable and reinforced by learning (or training). After finishing the learning, u_S cells can extract features from the input pattern. In other words, a u_S cell is activated only when a particular feature is presented at a certain position in the input layer. The features extracted by the u_S cells are determined during the learning process. Generally speaking, local features, such as a line at a particular orientation, are extracted in the lower

stages. More global features, such as part of a training pattern, are extracted in higher stages. The process of learning and the mechanism of feature extraction by u_S cells are discussed below in "Self-organization of the network."

The u_C cells are inserted in the network to allow for positional errors in the features of the stimulus. Connections from u_S cells to u_C cells are fixed and invariable.

The lower part of Figure 6 shows a detailed structure between layers of u_S cells and u_C cells. Each layer of u_S cells or u_C cells is divided into subgroups according to the features to which they respond.

The cells in each subgroup are arranged in a two-dimensional array. The connections converging to the cells in a subgroup are homogeneous: All the cells in a subgroup receive input connections of the same spatial distribution, where only the positions of the preceding cells shift in parallel with the position of the cells in the subgroup. This condition of homogeneity holds for fixed connections and for variable connections. As discussed in "Self-organization of the network," the reinforcement of the variable connections is always performed under this condition.

Each u_C cell receives signals from a group of u_S cells that extract the same feature, but from slightly different positions. The u_C cell is activated if at least one of these u_S cells is active. Even if the stimulus feature shifts position and another u_S cell is activated instead of the first one, the same u_C cell keeps responding. Hence, the u_C cell's response is less sensitive to shifts in position of the input pattern.

In the whole network, with its alternate layers of u_S cells and u_C cells, the process of feature extraction by u_S cells and toleration of positional shift by u_C cells repeats. During this process, local features extracted in a lower stage are gradually integrated into more global features. Figure 6 illustrates this situation.

Finally, each u_C cell of the recognition layer U_{CL} at the highest stage (denoted by u_{CL}) integrates all the information of the input pattern; each cell responds only to one specific pattern. In other words, only one u_{CL} cell, corresponding to the category of the input pattern, is activated. Other cells respond to the patterns of other categories.

Tolerating positional error a little at a

time at each stage, rather than all in one step, plays an important role in endowing the network with an ability to recognize even distorted patterns. Figure 7 illustrates this. Let a u_s cell in an intermediate stage of the network have already been trained to extract a global feature consisting of three local features of a training pattern "A" as shown in Figure 7a. The cell tolerates the positional error of each local feature, if its deviation falls within the dotted circle. Hence, this u_s cell responds to any of the deformed patterns shown in Figure 7b. The toleration of positional errors should not be too large at this stage. If too large errors are tolerated at one step, the network may come to respond erroneously, such as recognizing a stimulus like Figure 7c as an "A" pattern.

Since errors in the relative position of local features are tolerated in the process of extracting and integrating features, the same u_{CL} cell responds in the highest stage, even if the input pattern is deformed, changed in size, or shifted in position.* The network recognizes the "shape" of the pattern independent of its size and position.

When two or more patterns are simultaneously presented to the input layer U_{C0} , two or more u_{CL} cells may be activated at first. However, all of these cells but one soon stop responding. Usually only one u_{CL} cell continues to respond because of competition by lateral inhibition between feature-extracting u_s cells.² Lateral inhibition works in the highest stage and in the intermediate stages.

Self-organization of the network

The self-organization of the network results from learning-without-a-teacher. The network by itself acquires the ability to classify and to recognize patterns correctly on the basis of similarity in shape.

In the initial state before learning, all the variable connections in the network have

*It is difficult to state quantitatively to what degree the network can cope with deformation in patterns, because we do not have an appropriate mathematical measure to correctly express the psychological feeling of the deformation. We can get a rough idea of the degree of deformation that can be tolerated. For example, in an article to be published in April 1988,¹² Figure 14 shows some examples of deformed patterns that the neocognitron recognized correctly. These examples were obtained by a neocognitron trained by learning-with-a-teacher. Generally, the results obtained by learning-without-a-teacher are somewhat worse.

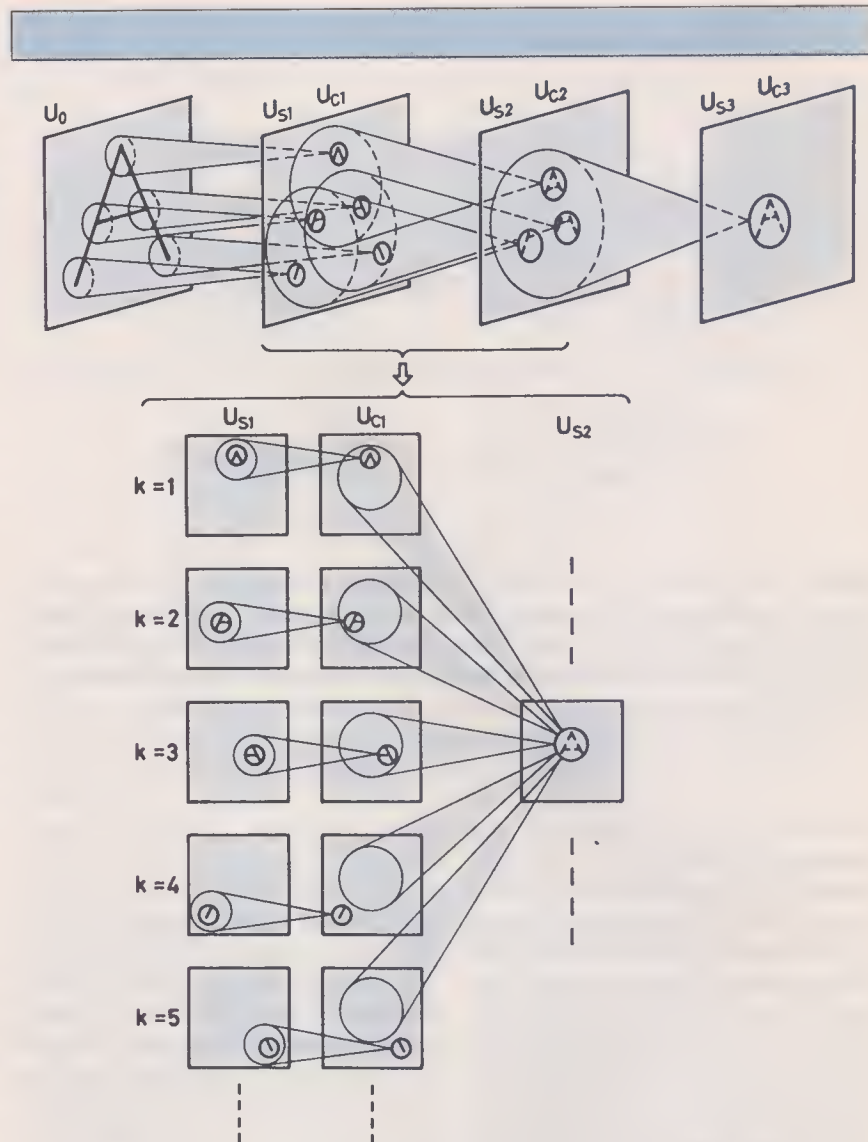


Figure 6. Illustration of the mechanism of pattern recognition in the forward paths.⁶

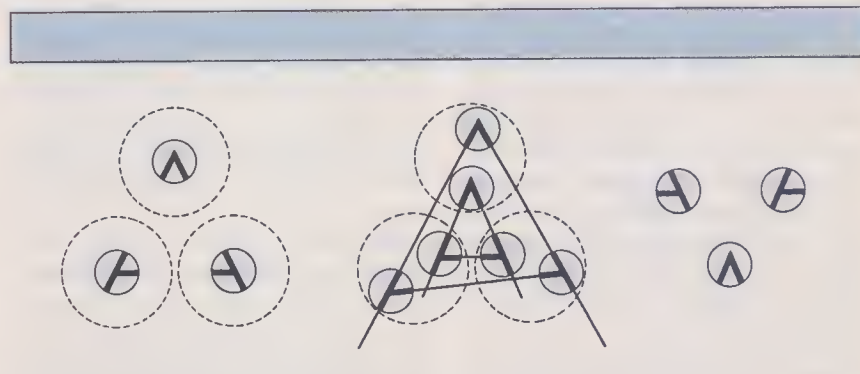


Figure 7. Illustration of the principle for recognizing deformed patterns.

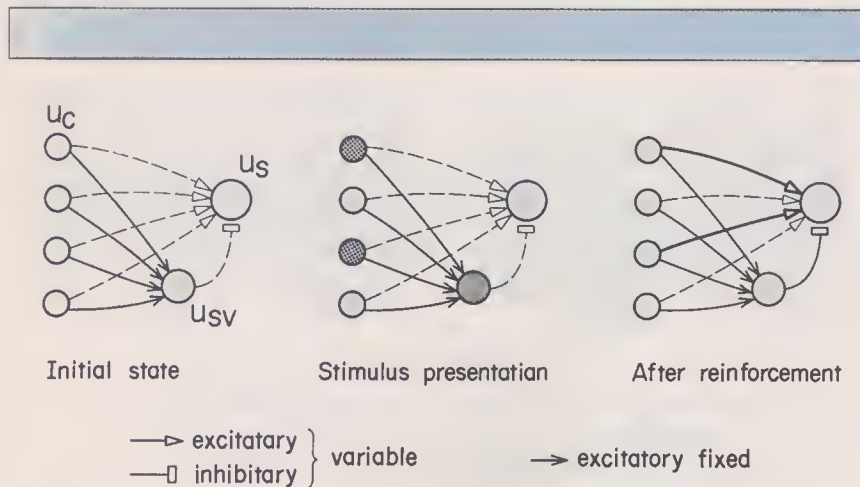


Figure 8. The process of reinforcement of the forward connections converging to a feature-extracting u_s cell. The density of the shadow in a circle represents the intensity of the response of the cell. (a) shows the initial state before training. (b) shows stimulus presentation during the training. (c) shows the connections after reinforcement.

a strength of zero. In the network, reinforcement of the forward connections comes first; the backward connections are reinforced later. Hence, during the process of reinforcement of the forward connections, no backward signal flows in the network. The variable forward connections are reinforced in a manner similar to that in the neocognitron.^{6,7} After finishing the reinforcement of the forward connections, the network reinforces the backward connections by the same amount as the forward connections with which they form pairs.

The reinforcement of the forward connections is performed according to two principles. The first was introduced for the self-organization of the cognitron⁴ model. Specifically, among the cells situated in a certain small area, only the one responding most strongly has its input connections reinforced. The amount of reinforcement of each input connection to this maximum-output cell is proportional to the intensity of the response of the cell from which the relevant connection leads.

This principle is applied to the variable input connections converging to feature-extracting u_s cells. Both excitatory and inhibitory connections are reinforced following this principle.

Figure 8 illustrates this process of reinforcement, showing only the forward connections converging to a u_s cell. As shown

in Figure 8a, the u_s cell receives variable excitatory connections from a group of u_C cells in the preceding layer. The cell also receives a variable inhibitory connection from a subsidiary inhibitory cell, called a u_{sv} cell. The u_{sv} cell receives fixed excitatory connections from the same group of u_C cells as does this u_s cell, and always responds with the average intensity of the output of the u_C cells. The initial strength of these variable connections is nearly zero.*

Suppose this u_s cell responds most strongly of the u_s cells in its vicinity when a training stimulus is presented (see Figure 8b). According to the first principle, variable connections leading from activated u_C and u_{sv} cells are reinforced as shown in Figure 8c. The variable excitatory connections to the u_s cell grow into a "template" that exactly matches the spatial distribution of the response of the cells in the preceding layer. The inhibitory variable connections from the u_{sv} cell are reinforced at the same time, but not as strongly because the output of the u_{sv} cell is not as large.

After completion of the training, the u_s cell acquires the ability to extract the fea-

*Each u_s cell has very weak and diffused excitatory connections only during the initial period of self-organization. Once a reinforcement of the input connections begins, these weak and diffuse initial connections disappear.¹²

ture of the stimulus presented during the training period. Through the excitatory connections, the u_s cell receives signals indicating the existence of the relevant feature to be extracted. If an irrelevant feature is presented, the inhibitory signal from the u_{sv} cell becomes stronger than the direct excitatory signals from the u_C cells, and the response of the u_s cell is suppressed. The u_s cell is activated only when the relevant feature is presented. We could say that the u_{sv} cell watches for the existence of irrelevant features. Thus, inhibitory u_{sv} cells play an important role in endowing the feature-extracting u_s cells with the ability to differentiate irrelevant features, and in increasing the selectivity of feature extraction.

According to this principle, among the u_s cells in a certain small area, only the one cell that yields the maximum output has its input connections reinforced. Because of the "winner-takes-all" nature of this principle, duplicate formation of cells that extract the same feature does not occur, and the formation of a redundant network can be prevented. Only the one cell giving the best response to a training stimulus is selected, and only that cell is reinforced so as to respond more appropriately to the stimulus.

Once a cell is selected and reinforced to respond to a feature, the cell usually loses its responsiveness to other features. When a different feature is presented, usually a different cell yields the maximum output and has its input connections reinforced. Thus, "division of labor" among the cells occurs automatically.

With this principle, the network also develops a self-repairing function. If a cell that has responded strongly to a stimulus is damaged and ceases to respond, another cell, which happens to respond more strongly than others, starts to grow and substitutes for the damaged cell. Until then, the larger response of the first cell had prevented the growth of a second cell.

The second principle introduced for the self-organization of the network states that the maximum-output cell not only grows, but also controls the growth of neighboring cells, working, so to speak, like a seed in crystal growth. Neighboring cells have their input connections reinforced in the same way as the seed cell.

When a seed cell is selected from a subgroup of u_s cells, all the other u_s cells in the subgroup grow to have input connections of the same spatial distribution as the seed cell. As a result, all the u_s cells in a subgroup grow to receive input connec-

tions of identical spatial distribution where only the positions of the preceding u_C cells have shifted in parallel with the positions of the u_S cells. Because connections develop iteratively in a subgroup, all the u_S cells in the subgroup come to respond selectively to a particular feature. Differences among these cells arise only from differences in position of the feature to be extracted.

Backward paths

The output of the recognition layer U_{CL} is sent back to lower stages through backward paths. It reaches the recall layer W_{CO} at the lowest stage of the backward paths. The backward signals are transmitted retracing the same route as the forward signals, because the cells in the backward paths receive gate signals from the cells in the forward paths. Guided by the forward signal flow, the backward signals reach exactly the same positions at which the input pattern is presented.

Since the backward signals are sent back only from the activated u_{CL} cell, only the signal components corresponding to the recognized pattern reach the recall layer W_{CO} . Therefore, we can interpret the output of the recall layer as the result of segmentation, where only components relevant to a single pattern are selected from the stimulus. Even if the stimulus pattern now recognized is a deformed version of a training pattern, the deformed pattern is segmented and emerges with its deformed shape.

Let us consider this process in more detail. First, look at the backward signals from an arbitrary w_S cell to the w_C cells of the preceding stage (see Figure 5). The network is designed so that the strength of the variable backward connections is automatically controlled in the following manner: After finishing the reinforcement of the forward connections (refer to Figure 8), the backward connections descending from a w_S cell are automatically reinforced to have a strength proportional to the forward connections ascending to the u_S cell paired with the w_S cell. Consequently, if an excitatory forward path forms to a u_S cell from a u_C cell, an excitatory backward path forms automatically from the corresponding w_S cell to the corresponding w_C cell. This also holds for the inhibitory backward path via the subsidiary w_{SV} cell, which corresponds to the u_{SV} cell in the forward paths. Hence, depending on whether a u_S cell receives an over-

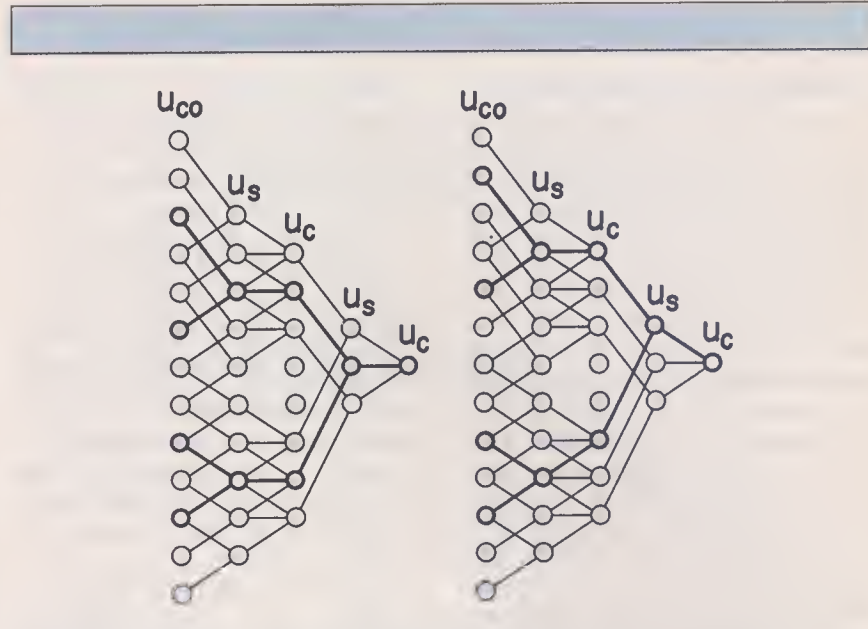


Figure 9. A simplified illustration of the forward signal flow in the network. Deformed stimuli (a) and (b) presented at different positions on the input layer (u_{CO}) elicit the same response from the u_C cell at the highest stage. The backward signals are controlled to retrace the same route as the forward signals, in the opposite direction.

all excitatory or inhibitory effect from a u_C cell through forward connections, the corresponding w_C cell also receives an overall excitatory or inhibitory effect from the corresponding w_S cell through backward connections.

Corresponding to the fixed forward connections that converge to a u_C cell from a number of u_S cells, many backward connections diverge from a w_C cell towards w_S cells (see Figure 5). However, we do not want all the w_S cells receiving excitatory backward signals from the w_C cell to be activated for the following reason: To activate a u_C cell in the forward path, the activation of at least one preceding u_S cell is sufficient. Usually only a small number of preceding u_S cells actually are activated, as shown in Figure 9. To elicit a similar response from the w_S cells in the backward paths, the network is synthesized in such a way that each w_S cell receives excitatory backward signals from w_C cells and a gate signal from the corresponding u_S cell. The w_S cell is activated only when it receives a signal both from u_S and w_C cells. Because of this network architecture, in the backward paths from w_C cells to w_S cells, the signals retrace the same route as the forward signals from u_S cells to u_C cells.

Gain control

Interaction between forward and backward signals is not unilateral. Forward cells receive gain-control signals from the corresponding backward cells, and the forward signal flow is facilitated by the backward signals. The gain of each u_C cell in forward paths is controlled by a signal from a corresponding w_C cell (Figures 4 and 5). When the w_C cell is silent, the gain between the inputs and the output of the u_C cell gradually attenuates from its initial value of 1.0 with the passage of time. When the w_C cell is activated, however, the attenuated gain recovers. Thus, only the forward signals are facilitated in the paths in which backward signals flow.

Now let's consider a case in which a stimulus consisting of two or more patterns is presented. Let one of the u_{CL} cells in the recognition layer be activated and one of the patterns in the stimulus recognized. Only the forward paths relevant to this pattern are facilitated by the action of backward signals from the u_{CL} cell. The forward paths corresponding to other patterns gradually lose their responsiveness because they receive no facilitation. Attention focuses selectively on only one of the patterns in the stimulus.

Threshold control

When some part of the input pattern is missing and the feature that should exist there fails to be extracted in the forward paths, the backward signal flow stops at that point. In such a case, the threshold for extraction of that feature automatically lowers and the model tries to extract even vague traces of the undetected feature. The w_{CX} cells detect the failure to extract a feature when the cells in the backward paths are active but the corresponding cells in the forward paths are not (Figures 4 and 5). The signal from w_{CX} cells weakens the efficiency of inhibition by u_{SV} cells and virtually lowers the threshold for feature extraction by u_S cells. (The signal from w_{CX} cells works like a neuromodulator in biological systems.) Thus, u_S cells are made to respond even to incomplete features to which they would not, in the normal state, respond.

Once a feature is thus extracted in the forward paths, the backward signal can be further transmitted to lower stages through the path unlocked by the newly activated forward cell. Hence, a complete pattern, including defective parts, emerges in the recall layer. Noise and blemishes have been eliminated from this pattern, because no backward signals return to their components. Thus, we can interpret the output of the recall layer W_{C0} as an auto-associative recall from associative memory.

Sometimes all the u_{CL} cells in the recognition layer are silent. The no-response state of the u_{CL} cells may occur, for instance, if the stimulus pattern differs greatly in shape from the original pattern, or if too many patterns are simultaneously presented to the input layer. If all the u_{CL} cells in the recognition layer are silent, information processing of the network goes no further because no backward signal flows in the network.

The no-response detector shown at the far right in Figure 4 always monitors the response of the u_{CL} cells. If all the u_{CL} cells are silent, the no-response detector sends another threshold-control signal through path x (shown in Figure 4) to all the u_S cells of all stages and lowers their threshold for feature extraction. The longer the silent state of the u_{CL} cells continues, the higher the value of the threshold-control signal. Hence, at least one u_{CL} cell is activated after a certain time.

Switching attention

Suppose one pattern in a composite stimulus is being attended to and recognized. A momentary interruption of the backward signal-flow suffices to switch attention to another pattern.

The gain of u_C cells is designed to be controlled as follows when switching attention: After the disappearance of the facilitating signal from the corresponding w_C cell in the backward path, each u_C cell has its gain lowered if the gain was previously kept high by facilitation. A decrease in gain occurs like fatigue, depending on the degree of the forced increase of the gain until then. On the other hand, the u_C cell will recover its gain if the gain was previously attenuated.

Because of this method of controlling the gain of the u_C cells, signals corresponding to the previous pattern have difficulty flowing through the forward paths. Usually another u_{CL} cell, hitherto silent, will be activated. (If no u_{CL} cell is activated, the no-response detector works until at least one u_{CL} cell is activated.) Consequently, the backward signals from the newly activated u_{CL} cell ease the flow of the forward signals for the new pattern. A repetition of this process switches attention to each of the patterns in the stimulus figure in turn, and they are recognized and recalled one after another.

Computer simulation

Let's look at the behavior of the model as simulated on a MicroVAX II minicomputer, with a program written in Fortran. The simulated network has three stages of hierarchy ($L = 3$). The input layer U_{C0} has 19×19 cells. The number of cells in the network totals about 41,399.*

The variable connections in the network were reinforced by learning-without-a-teacher. Figure 10 shows the five training patterns presented to the network during the learning period. Each of these patterns was presented only 11 times, enough to complete the self-organization.** During the learning period, these patterns were

*The number of cells actually used in the network depends on the set of training patterns presented during the training cycle. Because it is difficult to estimate in advance, this computer simulation series had a surplus. The total of 41,399, which excluded the cells in the non-response detector, included 19×19 u_{C0} and w_{C0} ; $19 \times 19 \times 21$ u_{S1} and w_{S1} ; 19×19 u_{SV1} and w_{SV1} ; $11 \times 11 \times 21$ u_{C1} , w_{C1} , and w_{CX1} ; $11 \times 11 \times 27$ u_{S2} , w_{S2} , u_{C2} , w_{C2} , and w_{CX2} ; 11×11 u_{SV2} and w_{SV2} ; $7 \times 7 \times 5$ u_{S3} and w_{S3} ; 7×7 u_{SV3} and w_{SV3} ; and $1 \times 1 \times 5$ u_{C3} .

repeatedly presented only in this shape; deformed versions were not presented at all. After completing the learning cycle, all the variable connections were fixed.

Figures 11 through 14 show the behavior of a network that has finished learning. In these figures, the response of the cells in the input layer U_{C0} and the recall layer W_{C0} is shown in time sequence. The numeral to the upper left of each pattern represents time t after the start of stimulus presentation.*** The stimulus pattern given to this network is identical to the response of the input layer at $t = 0$, shown in the upper left of each figure. (Note that the input pattern p appears directly in layer U_{C0} at $t = 0$, because no response is elicited from layer W_{C0} at $t < 0$.)

Figure 11 shows the response to a stimulus consisting of two juxtaposed patterns, "2" and "3." In the recognition layer, not shown in this figure, the u_{CL} cell corresponding to pattern "2" happens to be activated first. The signal is fed back to the recall layer through backward paths, but the middle part of the segmented pattern "2" is missing because of the interference from the closely adjacent "3." However, the missing part soon recovers because the components of pattern "3," which is not being attended to, are gradually attenuated by the decrease of gain of the forward cells.

At $t = 5$, the backward signal flow is interrupted for a moment to switch attention. The mark ▼ denotes the execution of this operation. Since the facilitating signals from backward cells stop, forward cells whose responsiveness has been kept high by facilitation will now lose some of their responsiveness. On the other hand, cells whose responsiveness has been decreased will recover their responsiveness. Thus, the forward paths for pattern "2," which have so far been facilitated, now lose their conductivity and the u_{CL} cell for pattern "3" is activated in the recognition layer. Since the backward signals are fed back from this newly activated u_{CL} cell, pattern "3" emerges in the recall layer.

Figure 12 shows an example of the response to a stimulus consisting of superimposed patterns. The pattern "4" is isolated first, the pattern "2" next, and finally pattern "1" is extracted.

***The reinforcement of the variable connections to cells of a higher stage was delayed until completion of the learning of the preceding stages. So, each of the training patterns was presented three, four, and four times for training stages 1, 2, and 3, respectively.

**Simulating the process of one step of t takes several seconds on a MicroVAX II, but computer time varies considerably depending on the presented stimulus.

Segmentation of individual patterns can be successful even if input patterns are deformed in shape or shifted in position. For example, "2" patterns in Figures 11 and 12 differ in shape from the training pattern (Figure 10), but the segmented patterns appearing in the recall layer are identical in shape to the stimulus patterns now presented.

Pattern segmentation can be successful even for a stimulus consisting of two patterns from the same category. Figure 13 shows such an example, in which the smaller "4" is segmented first, followed by the larger "4."

Figure 14 shows the response to a greatly deformed pattern with several parts missing and contaminated by noise. Because of the large difference between the stimulus and the training pattern, no response is elicited from the recognition layer (not shown in the figure) at first. Accordingly, no feedback signal appears at the recall layer W_{C0} . The no-response detector detects this situation, and the threshold-control signal is sent to all feature-extracting cells in the network, which makes them respond more easily even to incomplete features. Thus, at time $t=2$, the u_{CL} cell for "2" is activated in the recognition layer, and backward signals are fed back from it. In the pattern now sent back to the recall layer W_{C0} , noise has been completely eliminated, and some of the missing parts have begun to be interpolated. This partly interpolated signal, namely the output of the recall layer W_{C0} , is again fed back positively to the input layer U_{C0} . The interpolation continues gradually while the signal circulates through the feedback loop, and finally the missing part of the stimulus is completely filled in. The missing part is interpolated quite naturally, despite the considerable difference in shape between the stimulus and the training pattern.

In the pattern for which interpolation has already finished, the horizontal bar at the bottom of the "2" is shorter than in the training pattern. But no matter how short the horizontal bar, the pattern is a perfect character "2." Hence, this component of the pattern is left intact and is reproduced like the stimulus pattern. The deformation of the stimulus pattern is tolerated, and only indispensable missing parts are naturally interpolated, without strain. It is also important to note that noise has been completely eliminated, because attention is not focused on the components of the noise.

As we can see from these experiments,

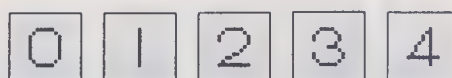


Figure 10. Five training patterns used for learning.

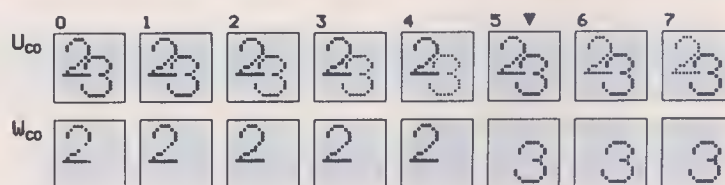


Figure 11. An example of the response to juxtaposed patterns.

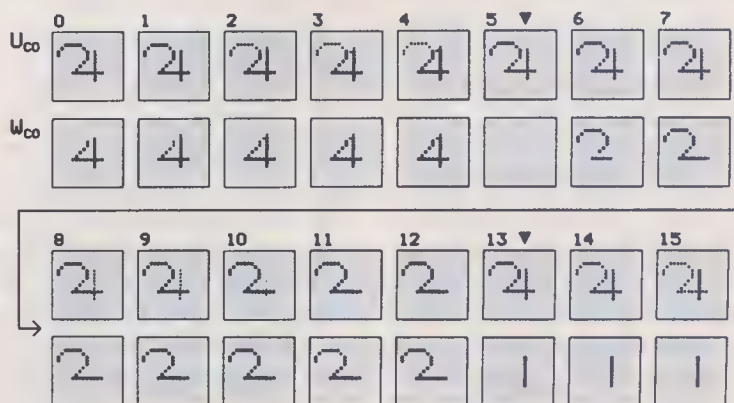


Figure 12. An example of the response to superimposed patterns.

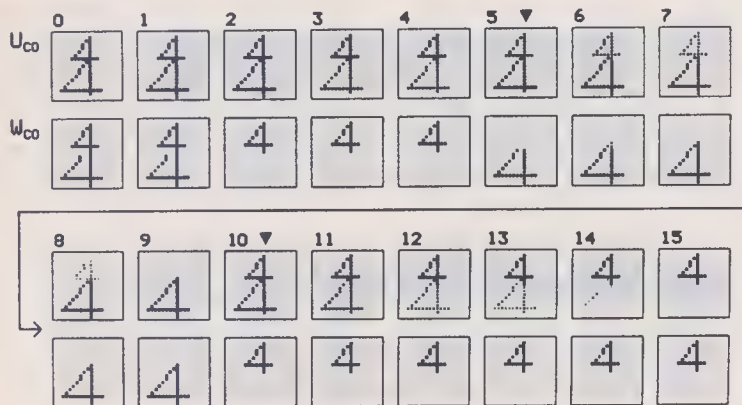


Figure 13. An example of the response to a stimulus consisting of two patterns of the same category.

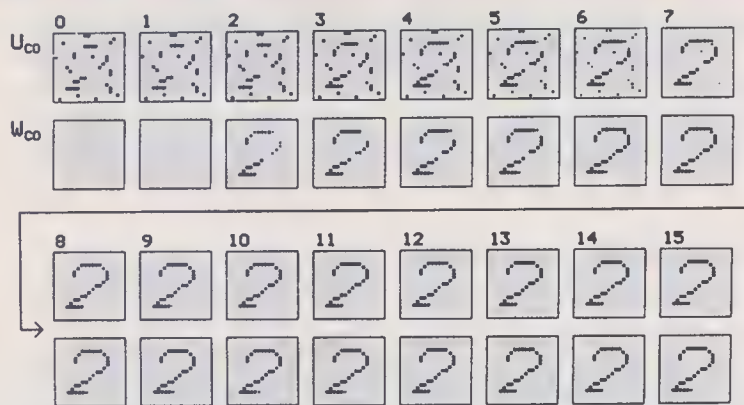


Figure 14. An example of the response to an incomplete distorted pattern with noise.

the model works well for most of the stimuli. However, the model has difficulty processing some stimuli, such as a pattern like "11" when drawn with an extremely narrow space between the two "1"s. Wide enough spacing poses no problem, but if the spacing is so narrow that the two parallel lines fall together into the receptive field of a single u_c cell of the first stage, the gain control of u_c cells may not work effectively to separate them.

The model discussed in this article has properties and abilities that most modern computers and pattern recognizers do not possess: pattern recognition, selective attention, segmentation, and associative recall. When a composite stimulus consisting of two or more patterns is presented, the model pays selective attention to each of the patterns one after the other, segments a pattern from the rest, and recognizes it separately.

We can also consider this model one of associative memory, with the ability to repair imperfect patterns. In contrast to earlier models, this model has perfect associative recall, even for deformed patterns, without regard to their positions.

Finally, the model can learn. We can train it to recognize any set of patterns. Hence, we can design a universal system that we can use, after training, for an individual purpose. □

Acknowledgments

The author would like to thank Takayuki Ito and Sei Miyake for their help on various aspects of this work.

References

1. K. Fukushima, "A Neural Network Model for Selective Attention in Visual Pattern Recognition," *Biological Cybernetics*, Vol. 55, No. 1, Oct. 1986, pp. 5-15.
2. K. Fukushima, "A Neural Network Model for Selective Attention in Visual Pattern Recognition and Associative Recall," *Applied Optics*, Vol. 26, No. 23, Nov. 1987, pp. 4985-4992.
3. D.E. Rumelhart, G.E. Hinton, and R.J. Williams, "Learning Representations by Back-Propagating Errors," *Nature*, Vol. 322, No. 6008, Oct. 1986, pp. 533-536.
4. K. Fukushima, "Cognitron: A Self-Organizing Multilayered Neural Network," *Biological Cybernetics*, Vol. 20, No. 3/4, Nov. 1975, pp. 121-136.
5. D.E. Rumelhart, and D. Zipser, "Feature Discovery by Competitive Learning," *Cognitive Science*, Vol. 9, 1985, pp. 75-112.
6. K. Fukushima, "Neocognitron: A Self-Organizing Neural Network Model for a Mechanism of Pattern Recognition Unaffected by Shift in Position," *Biological Cybernetics*, Vol. 36, No. 4, April 1980, pp. 193-202.
7. K. Fukushima and S. Miyake, "Neocognitron: A New Algorithm for Pattern Recognition Tolerant of Deformation and Shifts in Position," *Pattern Recognition*, Vol. 15, No. 6, 1982, pp. 455-469.
8. T. Kohonen, *Associative Memory: A System-Theoretical Approach*, Springer-Verlag, Berlin, 1977.
9. D.H. Hubel and T.N. Wiesel, "Receptive Fields, Binocular Interaction, and Functional Architecture in the Cat's Visual Cor-

tex," *J. Physiology*, Vol. 160, No. 1, London, 1962, pp. 106-154.

10. C. Bruce, R. Desimone, and C.G. Gross, "Visual Properties of Neurons in a Polysensory Area in the Superior Temporal Sulcus of the Macaque," *J. Neurophysiology*, Vol. 46, No. 2, Aug. 1981, pp. 369-384.
11. G.A. Carpenter and S. Grossberg, "A Massively Parallel Architecture for a Self-Organizing Neural Pattern Recognition Machine," *Computer Vision, Graphics and Image Processing*, Vol. 37, No. 1, Jan. 1987, pp. 54-115.
12. K. Fukushima, "Neocognitron: A Hierarchical Neural Network Capable of Visual Pattern Recognition," to appear in *Neural Networks*, Vol. 1, No. 2, April 1988.



Kunihiro Fukushima is a senior research scientist at the NHK (Japan Broadcasting Corp.) Science and Technical Research Laboratories. His major research interests concern the mechanism of information processing in the brain, and he is engaged in the synthesis of neural network models for the mechanisms of visual and auditory pattern recognition, selective attention, learning, self-organization, and memory.

Fukushima received a BS degree in electronics in 1958 and a PhD in electrical engineering in 1966, both from Kyoto University, Kyoto, Japan. He serves on the governing board of the International Neural Network Society.

Readers may write to Fukushima at NHK Science and Technical Research Laboratories, 1-10-11, Kinuta, Setagaya, Tokyo 157, Japan.

March 1988

COMPUTER SECURITY SCIENTISTS

Tomorrow's Computing
Technology is Today's Challenge

at
IDA

Some of the nation's most exciting developments in software technology, supercomputer architecture, AI, and expert systems are under scrutiny right now at the Institute for Defense Analyses. IDA is a Federally Funded Research and Development Center serving the Office of the Secretary of Defense, the Joint Chiefs of Staff, Defense Agencies, and other Federal sponsors.

IDA's Computer and Software Engineering Division (CSED) is seeking professional staff members with an in-depth theoretical and practical background in the area of Computer Security. Tasks include efforts on both the design/development of techniques to assess and assure security and providing advice to DoD decision makers on appropriate and feasible policy regarding security.

Specific desired skills and interests include:

- Formal verification, with emphasis on the Ada language
- Secure kernels and reference monitors
- Security in multiprocessor systems
- Fault-tolerance in secure systems
- Operating system, data base and network security criteria
- Testing and evaluation

Specialists in other areas of Computer Science are also sought: **Software Engineers, Distributed Systems, Artificial Intelligence and Expert Systems, and Programming Language Experts.**

We offer career opportunities at many levels of experience. You may be a highly experienced individual able to lead IDA projects and programs . . . or a recent MS/PhD graduate. You can expect a competitive salary, excellent benefits, and a superior professional environment. Equally important, you can expect a role on the leading edge of the state of the art in computing. If this kind of future appeals to you, we urge you to investigate a career with IDA. Please forward your resume to:

Mr. Thomas J. Shirhall
Manager of Professional Staffing
Institute for Defense Analyses
1801 N. Beauregard Street
Alexandria, VA 22311

An equal opportunity employer.
U.S. Citizenship is required.





Conference on Computer Performance Modelling,
Measurement and Evaluation

Santa Fe, New Mexico, USA
Conference May 25–27
Tutorials May 24

SPECIAL SPEAKER

Leonard Kleinrock, UCLA

SPECIAL SPEAKER

David DeWitt, Univ. of Wisconsin-Madison

TUTORIALS—May 24

Quantitative Operating Systems—Experimental Design

R. Bunt

UNIX: Requirements for Superscomputers

S. Bunch

The Other of the Two Cultures: Computer Performance
Measurements for Theoreticians

I. Bucher

Parallel Processing: A Look at Alternatives

D. Reed

Performance Evaluation of Multiprocessor Interconnection
Networks

L. Bhuyan

Seeing Through the Laplace Curtain: Numerical and
Approximate Methods for Laplace Transform Inversion

J. Abate and W. Whitt

Modular Simulation

M. Livny

Performance Analysis Using Petri Net Models

M. Vernan

Characterizing Distributed Systems Workloads

D. Ferrari

CPU Cache Memories

A. J. Smith

The Evaluation of Paging Algorithms in VM and MVS

T. Beretvas

Measurement of Computer System Reliability

R. Chillarege

SCHEDULED SESSIONS—May 25–27

Topics in Parallelism

Parallel Systems

Communication Networks

Load Sharing

Same Directions in Distributed Systems

Program Behavior and Paging

Queueing Network Modeling

Pastor Sessions

Techniques for Benchmarking Database Systems

Interconnection Networks

Case Studies

Multiprocessor Analysis

Measurement Techniques

Work in Progress

AUTHORS AND PARTICIPANTS

Agarwal, Alexander, Arai, Blake, Barn, Bruell, Bucher, Bunt, Chen, Chiu, Chlamtac, Capeland, Cavington, DeWitt, Eager, Ferrari, Ganz, Gupta, Haban, Haring, Harrison, Howe, Hang, Hsieh, Irgan, Jhunjhunwala, Jump, Kant, Keller, Kenevan, Kleinrock, Kastreva, Kathari, Kuba, Lam, Lazawska, Lee, Leutenegger, Lubachevsky, Lucier, Madala, Magrina, Majumdar, Mehta, Melus, Melvin, Mukherjee, Mukkamala, Murphy, Niemegeers, Patel, Patt, Pattipati, Reed, Reibman, Sanvicente, Shenker, Shinazaki, Shultz, Sinclair, Sudama, Tan, Tantawi, Teele, Tawsley, Tripathi, Trivedi, Vernan, Weinrib, Wolf, Wybraniec, Yashizawa, Zafiravic-Vukatic, Zaharjan

ORGANIZING COMMITTEE

Smith, Sevcik, Gaither, Keller, Williams, Bucher, Cabrera, Harris, Kaskela, Yantis

PROGRAM COMMITTEE

Balba, Bard, Bucher, Buzen, Cabrera, Howe, Heidelberger, Keller, Meyer, Neuse, Serazzi, Stewart, Tawsley, Tripathi, Zaharjan

ACCOMMODATIONS:

The El Darada Hotel offers first-class luxury accommodations in the Santa Fe historical district, 309 W. San Francisco St., Santa Fe, N.M. 87501 USA, (505) 988-4455. \$80 single/\$90 double.

REGISTRATION FEES:

(SIGMETRICS or ACM Member/Non-member)

	Before April 22	After April 22
Conference Plus Tutorials	\$375/435	\$450/510
Conference Only	245/295	265/310
Tutorials Only	145/155	205/215

Pre-registration brochures and complete
information can be obtained by contacting:

Rebecca Kaskela, SIGMETRICS 88

Mail Stop B265, LANL

Los Alamos, New Mexico 87545 USA

(505) 667-8887

Email: sigmetrics@lanl.gov

The ART of Adaptive Pattern Recognition by a Self-Organizing Neural Network

Gail A. Carpenter and Stephen Grossberg
Boston University

One of the central goals of computer science is to design intelligent machines capable of autonomous learning and skillful performance within complex environments that are not under strict external control. Many scientists have turned to a study of human capabilities as a source of new ideas for designing such machines. When a scientist undertakes such a study, he or she encounters a number of basic issues, of which we are all aware through our own personal experiences:

Why do we pay attention? Why do we learn expectations about the world? In particular, how do we cope so well with unexpected events? And, how do we manage to do as well as we do when we are on our own and do not have a teacher as a guide? How do we learn what combinations of facts are useful for dealing with a given situation, and what combinations of facts are irrelevant? How do we recognize familiar facts so quickly even though we have stored many other pieces of information? How do we combine knowledge about the external world with information about our internal needs to quickly make decisions that have a good chance of satisfying those needs? Finally, what do all of these properties have in common?

The adaptive resonance theory suggests a solution to the stability-plasticity dilemma facing designers of learning systems.

The stability-plasticity dilemma and ART

Researchers have found one answer to these questions through the attempt to solve a basic design problem, called the stability-plasticity dilemma, faced by all intelligent systems capable of autonomously adapting in real time to unexpected changes in their world. A developing the-

ory called adaptive resonance theory, or ART, suggests a solution to this problem.

The stability-plasticity dilemma asks: How can a learning system be designed to remain plastic, or adaptive, in response to significant events and yet remain stable in response to irrelevant events? How does the system know how to switch between its stable and its plastic modes to achieve stability without rigidity and plasticity without chaos? In particular, how can it preserve its previously learned knowledge while continuing to learn new things? And, what prevents the new learning from washing away the memories of prior learning?

We can easily dramatize the ubiquity of this problem: Imagine that you grew up in Boston before moving to Los Angeles, but periodically return to Boston to visit your parents. Although you may need to learn many new things to enjoy life in Los Angeles, these new learning experiences do not prevent you from remembering how to find your parent's house or otherwise get around Boston. A multitude of similar examples illustrate our ability to successfully adapt to environments where rules may change—without necessarily forgetting our old skills. Moreover, we are capable of successfully adapting to environments where rules may change

unpredictably, and we can do so even if no one tells us that the environment has changed. We can adapt, in short, without a teacher, through direct confrontation with our experiences. Such adaptation is called *self-organization* in the network modeling literature.

One of the key computational ideas rigorously demonstrated within the adaptive resonance theory is that top-down learned expectations focus attention upon bottom-up information in a way that protects previously learned memories from being washed away by new learning, and enables new learning to be automatically incorporated into the total knowledge base of the system in a globally self-consistent way.

The ART architectures discussed here are neural networks that self-organize stable recognition codes in real time in response to arbitrary sequences of input patterns. Within such an ART architecture, the process of adaptive pattern recognition is a special case of the more general cognitive process of hypothesis discovery, testing, search, classification, and learning. This property opens up the possibility of applying ART systems to more general problems of adaptively processing large abstract information sources and databases. This article outlines the main computational properties of these ART architectures, while comparing and contrasting these properties with those of alternative learning and recognition systems. Technical details are described in greater detail elsewhere,^{1,2} and several books collect articles in which the theory was developed through the analysis and prediction of interdisciplinary data about the brain and behavior.^{3,4}

Competitive learning models

ART models grew out of an analysis of a simpler type of adaptive pattern recognition network, often called a *competitive learning* model. Competitive learning models developed in the early 1970s through contributions of Christoph von der Malsburg⁵ and Stephen Grossberg,⁴ leading to the description of these models in 1976 in several forms in which they are used today.⁶ Authors such as Shun-ichi Amari,⁷ Leon Cooper,⁸ and Teuvo Kohonen⁹ have further developed these models. Kohonen⁹ has made particularly strong use of competitive learning in his

work on self-organizing maps. Grossberg⁴ has provided a historical discussion of the development of competitive learning models.

In a competitive learning model (see Figure 1), a stream of input patterns to a network F_1 can train the adaptive weights, or long-term memory (LTM) traces, that multiply the signals in the pathways from F_1 to a coding level F_2 . In the simplest such model, input patterns to F_1 are normalized before passing through the adaptive filter defined by the pathways from F_1 to F_2 . Level F_2 is designed as a competitive network capable of choosing the node which receives the largest total input ("winner-take-all"). The winning population then triggers associative pattern learning within the vector of LTM traces which sent its inputs through the adaptive filter.

For example, as in Figure 1, let I_i denote the input to the i th node v_i of F_1 , $i = 1, 2, \dots, M$; let x_i denote the activity, or short-term memory (STM) trace, of v_i ; let x_j denote the activity, or STM trace, of the j th node v_j of F_2 , $j = M + 1, \dots, N$; and let z_{ij} denote the adaptive weight, or long-term memory (LTM) trace, of the pathway from v_i to v_j . Then let

$$x_i = \frac{I_i}{\sum_{k=1}^M I_k} \quad (1)$$

be the normalized activity of v_i in response to the input pattern $I = (I_1, I_2, \dots, I_M)$. For simplicity, let the output signal S_i of v_i equal x_i . Let

$$T_j = \sum_{i=1}^M x_i z_{ij} \quad (2)$$

be the total signal received at v_j from F_1 , let

$$x_j = \begin{cases} 1 & \text{if } T_j > \max(T_k : k \neq j) \\ 0 & \text{if } T_j < \max(T_k : k \neq j) \end{cases} \quad (3)$$

summarize the fact that the node x_j in F_2 which receives the largest signal is chosen for short-term memory storage, and let a differential equation

$$\frac{d}{dt} z_{ij} = \epsilon x_j (-z_{ij} + x_i) \quad (4)$$

specify that only the vector $Z_j = (z_{1j}, z_{2j}, \dots, z_{Mj})$ of adaptive weights which about the winning node v_j is changed due to learning. Vector Z_j learns by reducing the

error between itself and the normalized vector $X = (x_1, x_2, \dots, x_M)$ in the direction of steepest descent.

Several equivalent ways describe how such a system recognizes input patterns I presented to F_1 . The winning node v_j in F_2 is said to code, classify, cluster, partition, compress, or orthogonalize these input patterns. In engineering, such a scheme is said to perform adaptive vector quantization. In cognitive psychology, it is said to perform categorical perception.³

In categorical perception, input patterns are classified into mutually exclusive recognition categories separated by sharp categorical boundaries. A sudden switch in pattern classification can occur if an input pattern is deformed so much that it crosses one of these boundaries and thereby causes a different node v_j to win the competition within F_2 . Categorical perception, in the strict sense of the word, occurs only if F_2 makes a choice. In more general competitive learning models, compressed but distributed recognition codes are generated by the model's coding level or levels.^{3,4,9}

In response to certain input environments, a competitive learning model possesses very appealing properties. It has been mathematically proved⁶ that, if not too many input patterns are presented to F_1 , or if the input patterns form not too many clusters, relative to the number of coding nodes in F_2 , then learning of the recognition code eventually stabilizes and the learning process elicits the best distribution of LTM traces consistent with the structure of the input environment.

Despite the demonstration of input environments that can be stably coded, it has also been shown, through explicit counterexamples,^{1,2,6} that a competitive learning model does not always learn a temporally stable code in response to an arbitrary input environment. In these counterexamples, as a list of input patterns perturbs level F_1 through time, the response of level F_2 to the *same* input pattern can differ on each successive presentation of that input pattern. Moreover, the F_2 response to a given input pattern might never settle down as learning proceeds.

Such unstable learning in response to a prescribed input is due to the learning that occurs in response to the other intervening inputs. In other words, the network's adaptability, or plasticity, enables prior learning to be washed away by more recent learning in response to a wide variety of input environments. In fact, infinitely

many input environments exist in which periodic presentation of just four input patterns can cause temporally unstable learning.^{1,2}

Learning can also become unstable due to simple changes in an input environment. Changes in the probabilities of inputs, or in the deterministic sequencing of inputs, can readily wash away prior learning. Moreover, this instability problem is not peculiar to competitive learning models. The problem is a basic one because it arises from a combination of the very features of an adaptive coding model that, on the surface, seem so desirable: its ability to learn from experience and its ability to code, compress, or categorize many patterns into a compact internal representation. Due to these properties, when a new input pattern I retrains a vector Z_j of LTM traces, the set of all input patterns coded by v_j also changes because a change in Z_j in Equation 2 can reverse the inequalities in Equation 3 in response to many of the input patterns previously coded by v_j .

Learning systems that can become unstable in response to many input environments cannot safely be used in autonomous machines that might be unexpectedly confronted by one of these environments on the job. Adaptive resonance theory was introduced in 1976 to show how to embed a competitive learning model into a *self-regulating control structure* whose autonomous learning and recognition proceed stably and efficiently in response to an arbitrary sequence of input patterns.

Self-stabilized learning in an arbitrary input environment

Figure 2 schematizes a typical example from a class of architectures called ART 1. It has been mathematically proved¹ that an ART 1 architecture is capable of stably learning a recognition code in response to an arbitrary sequence of binary input patterns until it utilizes its full memory capacity. Moreover, the adaptive weights, or LTM traces, of an ART 1 system oscillate at most once during learning in response to an arbitrary binary input sequence, yet do not get trapped in spurious memory states or local minima. After learning self-stabilizes, the input patterns directly activate the F_2 codes that best represent them.

As in a competitive learning model, an ART architecture encodes a new input pat-

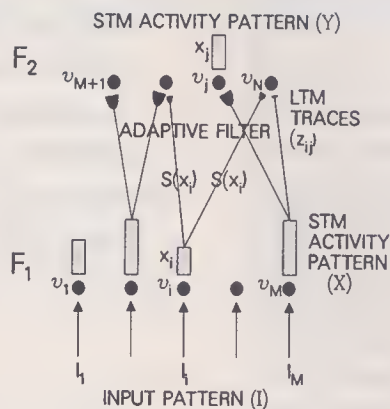


Figure 1. Stages of bottom-up activation: The input pattern I generates a pattern of STM activation $X = (x_1, x_2, \dots, x_M)$ across F_1 . Sufficiently active F_1 nodes emit bottom-up signals to F_2 . This signal pattern S is multiplied, or gated, by long-term memory (LTM) traces z_{ij} within the $F_1 \rightarrow F_2$ pathways. The LTM-gated signals are summed before activating their target nodes in F_2 . This LTM-gated and summed signal pattern T , where $T_j = \sum_i S_i z_{ji}$, generates a pattern of STM activation $Y = (x_{M+1}, \dots, x_N)$ across F_2 .

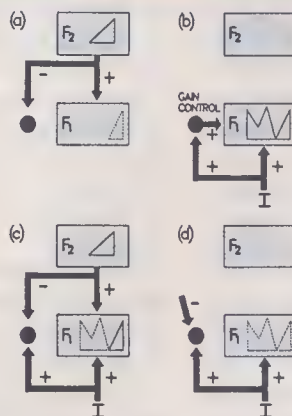


Figure 2. Matching by the 2/3 Rule: In (a), a top-down expectation from F_2 inhibits the attentional gain control source as it subliminally primes target F_1 cells. Dotted outline depicts primed activation pattern. In (b), only F_1 cells that receive bottom-up inputs and gain control signals can become supraliminally active. In (c), when a bottom-up input pattern and a top-down template are simultaneously active, only those F_1 cells that receive inputs from both sources can become supraliminally active. In (d), intermodality inhibition can shut off the F_1 gain control source and thereby prevent a bottom-up input from supraliminally activating F_1 , as when attention shifts to a different input channel. Similarly, disinhibition of the F_1 gain control source in (a) may cause a top-down prime to become supraliminal, as during an internally willed fantasy.

tern, in part, by changing the adaptive weights, or LTM traces, of a bottom-up adaptive filter. This filter is contained in the pathways leading from a feature representation field F_1 to a category representation field F_2 . In an ART network, however, a second, top-down adaptive filter, contained in the pathways from F_2 to F_1 , leads to the crucial property of code self-stabilization. Such top-down adaptive signals play the role of learned expectations in an ART system. Before considering details about how the ART control structure automatically stabilizes the learning process, we will sketch how self-stabilization occurs in intuitive terms.

Suppose that an input pattern I activates F_1 . Let F_1 in turn activate the code, or hypothesis, symbolized by the node v_{j1} at F_2 which receives the largest total signal from F_1 . Then, F_2 quickly reads out its learned top-down expectation to F_1 , whereupon the bottom-up input pattern and top-down learned expectation are matched across F_1 . If these patterns are badly matched, then a mismatch event takes place at F_1 which triggers a reset burst to F_2 . This reset burst shuts off node v_{j1} for the remainder of the coding cycle, and thereby deactivates the top-down expectation controlled by v_{j1} . Then, F_1 quickly reactivates essentially the same bottom-up signal pattern to F_2 as before. Level F_2 reinterprets this signal pattern, conditioned on the hypothesis that the earlier choice v_{j1} was incorrect, and another node v_{j2} is automatically chosen.

The parallel search, or hypothesis testing, cycle of bottom-up adaptive filtering from F_1 to F_2 , code (or hypothesis) selection at F_2 , read-out of a top-down learned expectation from F_2 to F_1 , matching at F_1 , and code reset at F_2 now repeats itself automatically at a very fast rate until one of three possibilities occurs: (1) a node v_{jm} is chosen whose top-down expectation approximately matches input I ; (2) a previously uncommitted F_2 node is selected; or (3) the full capacity of the system is used and cannot accommodate input I . Until one of these outcomes prevails, essentially no learning occurs, because all the STM computations of the hypothesis testing cycle proceed so quickly that the more slowly varying LTM traces in the bottom-up and top-down adaptive filters cannot change in response to them. Significant learning occurs in response to an input pattern only after the hypothesis testing cycle that it generates comes to an end.

If the hypothesis testing cycle ends in an approximate match, then the bottom-up

ART architectures differ from other popular neural network learning schemes in a number of basic ways.

input pattern and the top-down expectation quickly deform the activity pattern $X = (x_1, x_2, \dots, x_M)$ across F_1 into a net pattern that computes a fusion, or consensus, between the bottom-up and top-down information. This fused pattern represents the attentional focus of the system. When fusion occurs, the bottom-up and top-down signal patterns mutually reinforce each other via feedback and the system gets locked into a resonant state of STM activation. Only then can the LTM traces learn. What they learn is any new information about the input pattern represented within the fused activation pattern across F_1 . The fact that learning occurs only in the resonant state suggested the name "adaptive resonance theory." Thus, the system allows alteration of one of its prior learned codes only if an input pattern is sufficiently similar to what it already knows to risk a further refinement of its knowledge.

If the hypothesis testing cycle ends by selecting an uncommitted node at F_2 , then the bottom-up and top-down adaptive filters linked to this node learn the F_1 activation pattern generated directly by the input. No top-down alteration of the F_1 activation pattern occurs in this case. If the full capacity has been exhausted and no adequate match exists, learning is automatically inhibited.

In summary, an ART network refines its already learned codes based upon new information that can be safely accommodated into them via approximate matches, selects new nodes for initiating learning of novel recognition categories, or defends its fully committed memory capacity against being washed away by the incessant flux of new input events.

Alternative learning schemes

Many computational details have been worked out to make this scheme work well in an autonomous setting.^{1,2} Before we describe some of these details, we should note that ART architectures differ from other popular neural network learning schemes, such as autoassociators, the Boltzmann machine, and back propagation⁹⁻¹¹ in a number of basic ways. These differences are schematized in Table 1.

The most robust difference is that an ART architecture is designed to learn quickly and stably in real time in response to a possibly nonstationary world with an unlimited number of inputs until it utilizes its full memory capacity. Many alternative learning schemes become unstable unless they learn slowly in a controlled stationary environment with a carefully selected total number of inputs and do not use their full memory capacity.¹² For example, a learning system that is not self-stabilizing experiences a capacity catastrophe in response to an unlimited number of inputs: New learning washes away memories of prior learning if too many inputs perturb the system. To prevent this from happening, either the total number of input patterns that perturbs the system needs to be restricted, or the learning process itself must be shut off before the capacity catastrophe occurs.

Shutting off the world is not possible in many real-time applications. In particular, how can such a system allow a familiar input to be processed and recognized, but block the processing of a novel input pattern before the pattern destabilizes its prior learning? In the absence of a self-stabilization mechanism, an external teacher must act as the system's front end to independently recognize the inputs and make the decision. Shutting off learning at just the right time to prevent either a capacity catastrophe or a premature termination of learning would also require an external teacher. In either case, the external teacher must be able to carry out the recognition tasks that the learning system was supposed to carry out. Hence, non-self-stabilizing learning systems are not capable of functioning autonomously in ill-controlled environments.

In learning systems that need an external teacher to supply the correct representation to be learned, the learning process is often driven by mismatch between desired and actual outputs.^{10,11} Such

schemes must learn slowly, and in a stationary environment, or risk unstable oscillations in response to the mismatches. They can also be destabilized if the external teaching signal is noisy, because such noise creates spurious mismatches.

These learning models also tend to get trapped in local minima, or globally incorrect solutions. Models such as simulated annealing and the Boltzmann machine¹⁰ use internal system noise to escape local minima and approach a more global minimum. An externally controlled (temperature) parameter regulates this process by making it converge ever more slowly to a critical value.

In contrast, approximate matches, rather than mismatches, drive the learning process in ART. Learning in the approximate-match mode enables rapid and stable learning to occur while buffering the system's memory against external noise. The hypothesis testing cycle replaces internal system noise as a scheme for discovering a globally correct solution. It does not use an externally controlled temperature parameter or teacher.

Matching by the 2/3 rule

One of the key constraints on the design of the ART 1 architecture is its rule for matching a bottom-up input pattern with a top-down expectation at F_1 . This rule, called the 2/3 Rule,¹ is necessary to regulate both the hypothesis testing cycle and the self-stabilization of learning in an ART 1 system.

The 2/3 Rule reconciles two properties whose simplicity tends to conceal their fundamental nature: In response to an arbitrary bottom-up input pattern, F_1 nodes can be *supraliminally* activated; that is, activated enough to generate output signals to other parts of the network and thereby to initiate the hypothesis testing cycle. In response to an arbitrary top-down expectation, however, F_1 nodes are only *subliminally* activated; they sensitize, prepare, or attentionally *prime* F_1 for future input patterns that may or may not generate an approximate match with this expectation, but do not, in themselves, generate output signals. Such a subliminal reaction enables an ART system to anticipate future events and thus to function as an "intentional" machine. In particular, if an attentional prime is locked into place by a high-gain top-down signal source, then an ART system can automatically

Table 1. ART architectures compared to other learning schemes.

ART architecture	Alternative learning properties
Real-time (on-line) learning	Lab-time (off-line) learning
Nonstationary world	Stationary world
Self-organizing (unsupervised)	Teacher supplies correct answer (supervised)
Memory self-stabilizes in response to arbitrarily many inputs	Capacity catastrophe in response to arbitrarily many inputs
Effective use of full memory capacity	Can only use partial memory capacity
Maintain plasticity in an unexpected world	Externally shut off plasticity to prevent capacity catastrophe
Learn internal top-down expectations	Externally impose costs
Active attentional focus regulates learning	Passive learning
Slow or fast learning	Slow learning or oscillation catastrophe
Learn in approximate-match phase	Learn in mismatch phase
Use self-regulating hypothesis testing to globally reorganize the energy landscape	Use noise to perturb system out of local minima in a fixed energy landscape
Fast adaptive search for best match	Search tree
Rapid direct access to codes of familiar events	Recognition time increases with code complexity
Variable error criterion (vigilance parameter) sets coarseness of recognition code in response to environmental feedback	Fixed error criterion in response to environmental feedback
All properties scale to arbitrarily large system capacities	Key properties deteriorate as system capacity increased

suppress all inputs that do not fall into a sought-after recognition category, yet amplify and hasten the processing of all inputs that do.³

To implement the 2/3 Rule, we need to assure that F_1 can distinguish between bottom-up and top-down signals, so that it can supraliminally react to the former and subliminally react to the latter. In ART 1, this distinction is determined by a third F_1 input source, called an *attentional gain control* channel, that responds differently to bottom-up and top-down signals.

Figure 2 describes how this gain control source works. When activated, it excites each F_1 node equally. The 2/3 Rule says that at least two out of three input sources are needed to supraliminally activate an

F_1 node; the three are a bottom-up input, a top-down input, and a gain control input. In the top-down processing mode (see Figure 2a), each F_1 node receives a signal from at most one input source and, hence, is only subliminally activated. In the bottom-up processing mode (Figure 2b), each active bottom-up pathway can turn on the gain control node, whose output, once on, is independent of the total number of active bottom-up pathways. Then, all F_1 nodes receive at least a gain control input, but only those nodes that also receive a bottom-up input are supraliminally activated.

When both bottom-up and top-down inputs reach F_1 (see Figure 2c), the gain control source is shut off, so that only those F_1 nodes which receive top-down

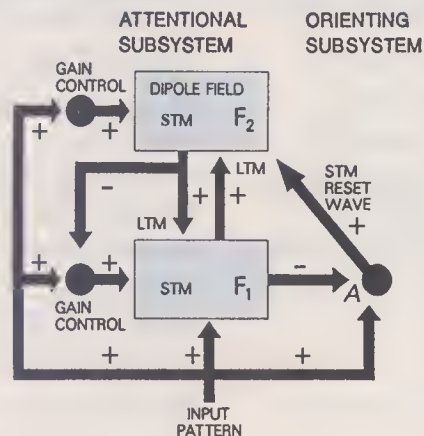


Figure 3. ART 1 system: Two successive stages, F_1 and F_2 , of the attentional subsystem encode patterns of activation in short-term memory (STM). Bottom-up and top-down pathways between F_1 and F_2 contain adaptive long-term memory (LTM) traces which multiply the signals in these pathways. The remainder of the circuit modulates these STM and LTM processes. Modulation by gain control enables F_1 to distinguish between bottom-up input patterns and top-down priming, or expectation, patterns, and to match these bottom-up and top-down patterns by the 2/3 Rule. Gain control signals also enable F_2 to react supraliminally to signals from F_1 while an input pattern is on. The orienting subsystem generates a reset wave to F_2 when sufficiently large mismatches between bottom-up and top-down patterns occur at F_1 . This reset wave selectively and enduringly inhibits previously active F_2 cells until the input is shut off.

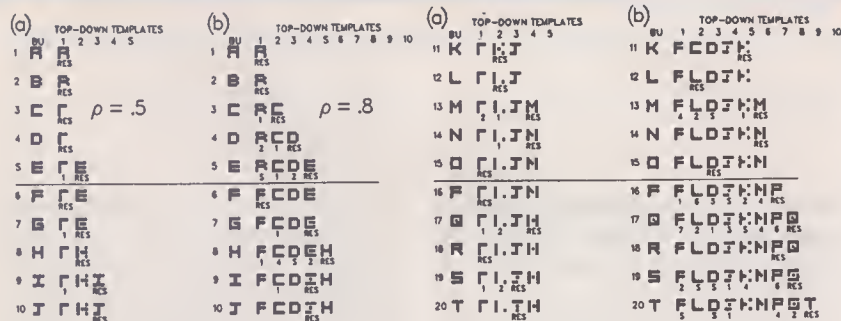


Figure 4. Alphabet learning: Code learning by ART 1 in response to the first presentation of the first 20 letters of the alphabet is shown. Two different vigilance levels were used, $\rho = .5$ and $\rho = .8$. Each row represents the total code learned after the letter at the left-hand column of the row is presented at F_1 . Each column represents the learning, through time, of the top-down LTM vector, or expectation, corresponding to the F_2 node whose index is listed at the top of the column. These LTM vectors do not, in general, equal the input patterns which change them through learning. Instead, each expectation acts like a novel type of prototype for the entire set of practiced input patterns coded by that node, as well as for unfamiliar input patterns that share invariant properties with this set. The simulation illustrates the "fast learning" case, in which the altered LTM traces reach a new equilibrium in response to each new stimulus. Slow learning is more gradual than this.

confirmation of the bottom-up input are supraliminally activated. In this case, the 2/3 Rule maintains supraliminal activity only within the spatial intersection of the bottom-up input pattern and the top-down expectation. Consequently, if a bottom-up input pattern, as in Figure 2b, causes the read-out of a badly matched top-down expectation, as in Figure 2c, then the total number of supraliminally active F_1 nodes can suddenly decrease, thereby causing a decrease in the total output signal emitted by F_1 . This property is used heavily in controlling the hypothesis testing and self-stabilization processes, as we will show next.

Automatic control of hypothesis testing

An ART architecture automates its hypothesis testing cycle through interactions between an attentional subsystem and an orienting subsystem. These subsystems in the ART 1 architecture are schematized in Figure 3.

The orienting subsystem A generates an output signal only when a mismatch occurs between a bottom-up input pattern and top-down expectation at level F_1 of the attentional subsystem. Thus, A functions like a novelty detector. The output signal from A is called an *STM reset wave* because it selectively inhibits the active node(s) at level F_2 of the attentional subsystem. The novelty detector A thereby disconfirms the F_2 hypothesis that led to the F_1 mismatch.

The 2/3 Rule controls the reset wave emitted by A as follows: When a bottom-up input pattern is presented, each of the active input pathways to F_1 also sends a signal to the orienting subsystem A , where all of these signals are added up. When the input pattern activates F_1 , each of the activated F_1 nodes sends an inhibitory signal to A . The system is designed so that the total inhibitory signal is larger than the total excitatory signal. Thus, in the bottom-up mode, the balance between active F_1 nodes and active input lines prevents a reset wave from being triggered. (Note that level F_1 in ART 1 is not normalized as it was in Equation 1 of the competitive learning model and in the ART 2 systems discussed below. The decision of whether and how to normalize depends upon the design of the whole system.)

This balance is upset when a top-down expectation is read out that mismatches the bottom-up input pattern at F_1 . As in Fig-

ure 2c, the total output from F_1 then decreases by an amount that grows with the severity of the mismatch. If the attenuation is sufficiently great, then inhibition from F_1 to A can no longer prevent A from emitting a reset wave. A parameter ρ called the *vigilance parameter* determines how large a mismatch will be tolerated before A emits a reset wave. High vigilance forces the system to search for new categories in response to small differences between input and expectation. Then, the system learns to classify input patterns into a large number of fine categories. Low vigilance enables the system to tolerate large mismatches and thus group together input patterns according to a coarse measure of mutual similarity. The vigilance parameter may be placed under external control, being increased, for example, when the network is "punished" for failing to distinguish two inputs that give rise to different consequences.^{1,3}

Figure 4 schematizes learning in response to the first 20 input presentations of a computer simulation of alphabet learning. After presenting the 20th input, nine recognition categories have formed when $\rho = .8$, but only four categories have been formed when $\rho = .5$. In this computer experiment, learning self-stabilized after at most three presentations of the 26 letters at any level of vigilance, and the learned LTM codes were more abstract—that is, less letter-like—at lower levels of vigilance.

Figure 5 illustrates how these properties of the interaction between levels F_1 , F_2 , and A regulate the hypothesis testing cycle of the ART 1 system. In Figure 5a, an input pattern I generates an STM activity pattern X across F_1 . The input pattern I also excites the orienting subsystem A , but pattern X at F_1 inhibits A before it can generate an output signal. Activity pattern X also elicits an output pattern S which activates the bottom-up adaptive filter $T = ZS$, where Z is the matrix of bottom-up LTM traces. As a result, an STM pattern Y becomes active at F_2 . In Figure 5b, pattern Y generates a top-down output U through the adaptive filter $V = \hat{Z}U$, where $\hat{Z}$ is the matrix of top-down LTM traces. Vector V is the top-down expectation read into F_1 . Expectation V mismatches input I , significantly inhibiting STM activity across F_1 . The amount by which activity in X is attenuated to generate the activity pattern X^* depends upon how much of the input pattern I is encoded within the expectation V , via the 2/3 Rule.

When a mismatch attenuates STM

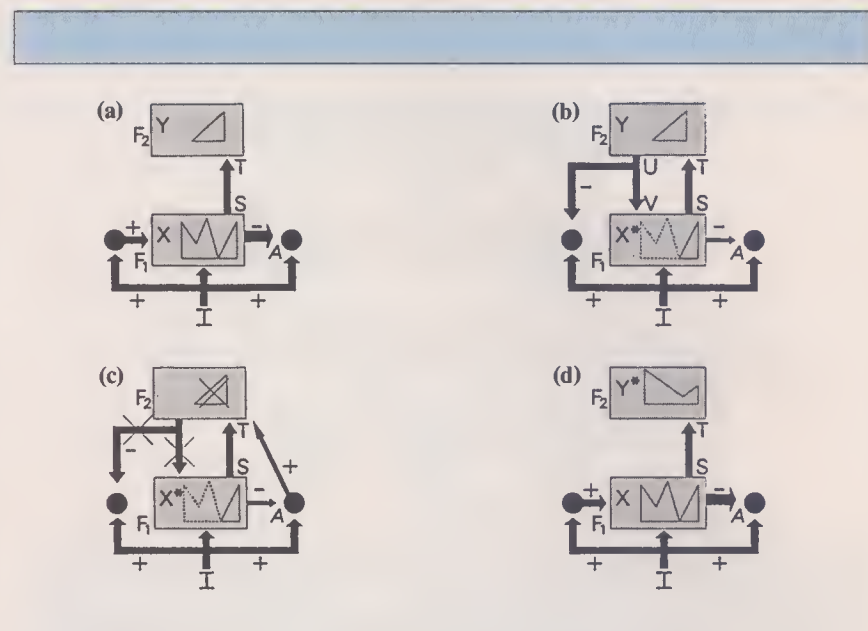


Figure 5. ART 1 hypothesis testing cycle: In (a), the input pattern I generates the STM activity pattern X at F_1 as it activates A . Pattern X both inhibits A and generates the bottom-up signal pattern S . Signal pattern S is transformed via the adaptive filter into the input pattern $T = ZS$, which activates the compressed STM pattern Y across F_2 . In (b), pattern Y generates the top-down signal pattern U which is transformed by the top-down adaptive filter $V = \hat{Z}U$ into the expectation pattern V . If V mismatches I at F_1 , then a new STM activity pattern X^* is generated at F_1 . The reduction in total STM activity that occurs when X is transformed into X^* causes a decrease in the total inhibition from F_1 to A . In (c), then, the input-driven activation of A can release a nonspecific arousal wave to F_2 , which resets the STM pattern Y at F_2 . In (d), after Y is inhibited, its top-down expectation is eliminated, and X can be reinstated at F_1 . Now X once again generates input pattern T to F_2 , but since Y remains inhibited T can activate a different STM pattern Y^* at F_2 . If the top-down expectation due to Y^* also mismatches I at F_1 , then the rapid search for an appropriate F_2 code continues.

activity across F_1 , the total size of the inhibitory signal from F_1 to A is also attenuated. If the attenuation is sufficiently great, inhibition from F_1 to A can no longer prevent the arousal source A from firing. Figure 5c depicts how disinhibition of A releases an arousal burst to F_2 which equally, or nonspecifically, excites all the F_2 cells. The cell populations of F_2 react to such an arousal signal in a state-dependent fashion. In the special case that F_2 chooses a single population for STM storage, the arousal burst selectively inhibits, or resets, the active population in F_2 . This inhibition is long-lasting.

In Figure 5c, inhibition of Y leads to removal of the top-down expectation V , and thereby terminates the mismatch

between I and V . Input pattern I can thus reinstate the original activity pattern X across F_1 , which again generates the output pattern S from F_1 and the input pattern T to F_2 . Due to the enduring inhibition at F_2 , the input pattern T can no longer activate the original pattern Y at F_2 . Level F_2 has been conditioned by the disconfirmation of the original hypothesis. A new pattern Y^* is thus generated at F_2 by I (see Figure 5d).

The new activity pattern Y^* reads out a new top-down expectation V^* . If a mismatch again occurs at F_1 , the orienting subsystem is again engaged, leading to another arousal-mediated reset of STM at F_2 . In this way, a rapid series of STM matching and reset events may occur. Such an STM matching and reset series controls

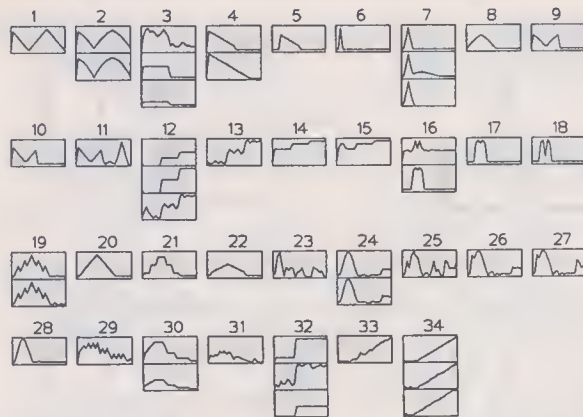


Figure 6. Category grouping by ART 2 of 50 analog input patterns into 34 recognition categories. Each input pattern I is depicted by a graph as a function of abscissa values i ($i = 1 \dots M$), with successive ordinate I_i values connected by straight lines. The category structure established upon one complete presentation of the 50 inputs thereafter remains stable if the same inputs are presented again.

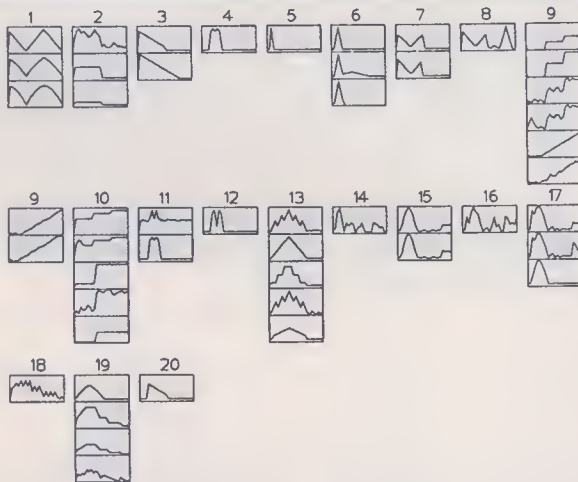


Figure 7. Lower vigilance implies coarser grouping. The same ART 2 system as used in Figure 6 has here grouped the same 50 inputs into 20 recognition categories. Note, for example, that Categories 1 and 2 of Figure 6 are joined in Category 1; Categories 14, 15, and 32 are joined in Category 10; and Categories 19-22 are joined in Category 13.

the system's hypothesis testing and search of LTM by sequentially engaging the novelty-sensitive orienting subsystem.

Although STM is reset sequentially in time by this mismatch-mediated, self-terminating LTM search process, the mechanisms that control the LTM search are all parallel network interactions, rather than serial algorithms. Such a parallel search scheme continuously adjusts itself to the system's evolving LTM codes. The

LTM code depends on both the system's initial configuration and its unique learning history, and hence cannot be predicted by a prewired search algorithm. Instead, the mismatch-mediated engagement of the orienting subsystem triggers a process of parallel self-adjusting search that tests only the hypotheses most likely to succeed, given the system's unique learning history.

The mismatch-mediated search of LTM ends when an STM pattern across F_2

reads out a top-down expectation that approximately matches I (to the degree of accuracy required by the level of attentional vigilance) or that has not yet undergone any prior learning. In the former case, the accessed recognition code is refined based on any novel information contained in the input I ; that is, based upon the activity pattern resonating at F_1 that fuses together bottom-up and top-down information according to the 2/3 Rule. In the latter case, a new recognition category is established as a new bottom-up code and top-down template are learned.

ART 2: Learning to recognize an analog world

Although self-organized recognition of binary patterns is useful in many applications, such as recognition of printed or written text, as in Figure 4, many other applications require the ability to categorize arbitrary sequences of analog (including binary) input patterns. A class of architectures, generically called ART 2, has been developed for this purpose.²

Given the enhanced capabilities of ART 2 architectures, a sequence of arbitrary input patterns can be fed through an arbitrary preprocessor before the output patterns of the preprocessor are fed as inputs into an ART 2 system for automatic classification. Figure 6 illustrates how an ART 2 architecture has quickly learned to stably classify 50 analog input patterns, chosen to challenge the architecture in multiple ways, into 34 recognition categories after a single learning trial. Figure 7 illustrates how the same 50 input patterns have been quickly classified into 20 coarser categories after a single learning trial, using a smaller setting of the vigilance parameter.

ART 2 architectures can autonomously classify arbitrary sequences of analog input patterns into categories of arbitrary coarseness while suppressing arbitrary levels of noise. They accomplish this by modifying the ART 1 architecture to incorporate solutions of several additional design problems into their circuitry. In particular, level F_1 is split into separate sublevels for receiving bottom-up input patterns, for receiving top-down expectations, and for matching the bottom-up and top-down data, as in Figure 8.

Three versions of the ART 2 architecture are now being applied to problems such as visual pattern recognition, speech

perception, and radar classification. In addition to research on ART undertaken at several universities, applications are also being developed at government laboratories and industrial firms including the MIT Lincoln Laboratory; Booz-Allen and Hamilton, Inc.; Hecht-Nielsen Neurocomputer Corp.; Science Applications International Corp.; the U.S. Army Research Center at Redstone Arsenal; and Wright-Patterson Air Force Base.

Invariant visual pattern recognition

Researchers from Boston University and the MIT Lincoln Laboratory are collaborating to carry out an application to invariant visual pattern recognition. This application uses a three-stage preprocessor, summarized in Figure 9.

First, the image figure to be recognized is detached from the image background using laser radar sensors. This can be accomplished by intersecting the images formed by two laser sensors: the image formed by a range detector focused at the distance of the figure and the image formed by another laser detector capable of differentiating figure from background, such as a doppler image when the figure is moving or the intensity of laser return when the figure is stationary.¹³

The second stage of the preprocessor contains a neural network, called a boundary contour system,^{3,4} that detects, sharpens, regularizes, and completes the boundaries within noisy images.

The third stage of the preprocessor contains a Fourier-Mellin filter, whose output spectra are invariant under such image transformations as 2D spatial translation, dilation, and rotation.¹⁴

Thus, the input patterns to ART 2 are the invariant spectra of completed boundary segmentations of laser radar sensors. By setting ART 2 parameters to suppress (up to) a prescribed level of input noise and to tolerate (up to) a prescribed level of input deformation, this system defines a compact circuit capable of autonomously learning to recognize visual targets that are deformed, rotated, dilated, and shifted. Although this preprocessor does not purport to provide a biological solution to the problem of invariant visual object recognition, we know that the mammalian visual cortex does carry out computations analogous to aspects of the second and third stages of this preprocessor.^{3,4,15}

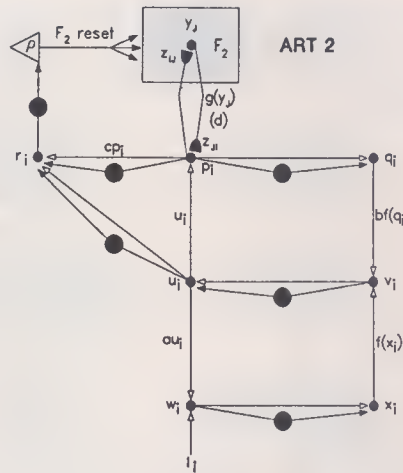


Figure 8. A typical ART 2 architecture. Open arrows indicate specific patterned inputs to target nodes. Filled arrows indicate nonspecific gain control inputs. The gain control nuclei (large filled circles) nonspecifically inhibit target nodes in proportion to the L_2 -norm of STM activity in their source fields. As in ART 1, gain control (not shown) coordinates STM processing with input presentation rate.

The three R's: Recognition, reinforcement, and recall

Recognition is only one of several processes whereby an intelligent system can learn a correct solution to a problem. Reinforcement and recall are no less important in designing an autonomous intelligent system.

Reinforcement, notably reward and punishment, provides additional information in the form of environmental feedback based on the success or failure of actions triggered by a recognition event. Reward and punishment calibrate whether the action has or has not satisfied internal needs, which in the biological case include hunger, thirst, sex, and pain reduction, but may in machine applications include a wide variety of internal cost functions. Reinforcement can modify the formation of recognition codes and can shift attention to focus upon those codes whose activation promises to satisfy internal needs based on past experience. For example, both green and yellow bananas may be recognized as part of a single recognition category until reinforcement signals, contingent upon eating the

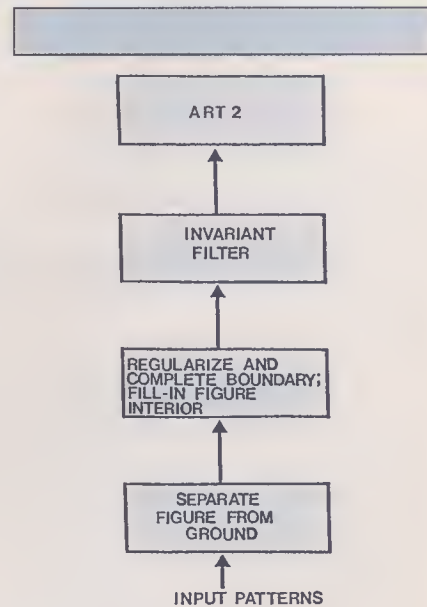


Figure 9. A three-stage preprocessor for the ART 2 system enables input patterns that are deformed, shifted, dilated, and rotated to be recognized as exemplars of the same category. The preprocessor passes laser radar images that separate figure from background through a boundary segmentation network and then through a Fourier-Mellin transform. The Fourier-Mellin spectra are the inputs to ART 2.

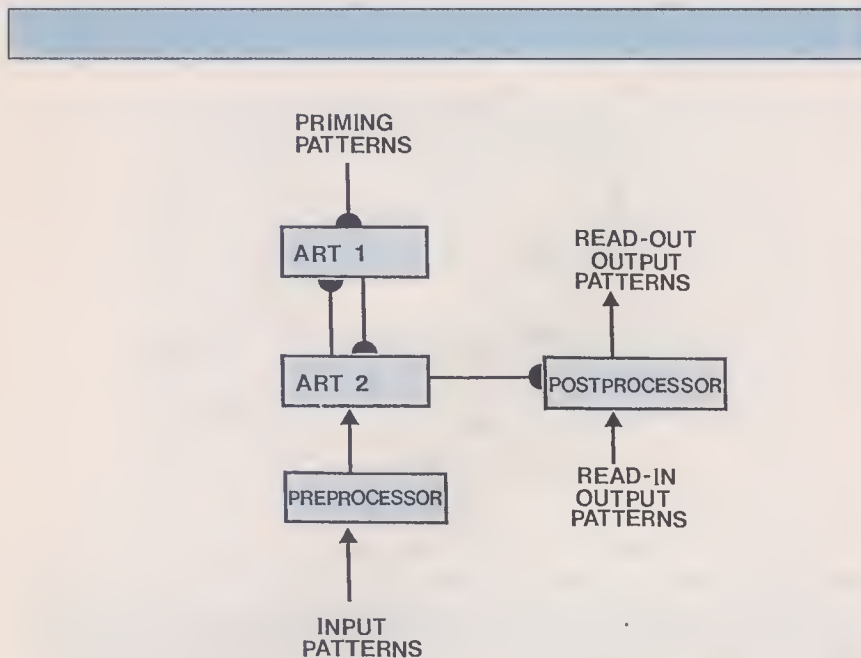


Figure 10. A self-organizing architecture for invariant pattern recognition and recall that can be expanded, as noted in the text, to include reinforcement mechanisms capable of focusing attention upon internally desired classes of external events.

bananas, differentiate them into separate categories.

Recall can generate equivalent responses or actions to input events classified by different recognition codes. For example, printed and script letters might generate distinct recognition codes, yet can also elicit identical learned naming responses.

Our own research program during the past two decades at Boston University has been devoted to discovering and implementing models of self-organizing biological systems wherein all the ingredients of recognition, reinforcement, and recall join together in a single integrated circuit.^{3,4} The system depicted in Figure 10 provides a framework for implementing some of these circuit designs. In particular, as ART 2 self-organizes recognition categories in response to the preprocessed inputs, its categorical choices at the F_2 classifying level self-stabilize through time. In examples wherein F_2 makes a choice, ART 2 can be used as the first level of an ART 1 architecture, or yet another ART 2 architecture. Let us call the classifying level of this latter architecture

F_3 . Level F_3 can be used as a source of pre-wired priming inputs to F_2 .

Alternatively, as in Figure 10, self-stabilizing choices by F_3 can quickly be learned in response to the choices made at F_2 . Then, F_3 can be used as a source of self-organized priming inputs to F_2 , and a source of priming patterns can be associated with each of the F_3 choices via mechanisms of associative pattern learning.³ After learning of these primes, turning on a particular prime can activate a learned $F_3 \rightarrow F_2$ top-down expectation. Then F_2 can be supraliminally activated only by an input exemplar which is a member of the recognition category of the primed F_2 node.

The architecture ignores all but the primed set of input patterns. In other words, the prime causes the architecture to pay attention only to expected sources of input information. Due to the spatial invariance properties of the preprocessor, the expected input patterns can be translated, dilated, or rotated in 2D without damaging recognition. Due to the similarity grouping properties of ART 2 at a fixed

level of vigilance, suitable deformations of these input patterns, including deformations due to no more than anticipated levels of noise, can also be recognized.

The output pathways from level F_2 of ART 2 to the postprocessor can learn to recall any spatial pattern or spatiotemporal pattern of outputs by applying theorems about associative learning in a type of circuit called an *avalanche*.³ In particular, distinct recognition categories can learn to generate identical recall responses. Thus, the architecture as a whole can stably self-organize an invariant recognition code and an associative map to an arbitrary format of output patterns.

The interactions (priming $\rightarrow$ ART) and (ART $\rightarrow$ postprocessor) in Figure 10 can be modified so that output patterns are read out only if the input patterns have yielded rewards in the past and if the machine's internal needs for these rewards have not yet been satisfied.^{3,4} In this variation of the architecture, the priming patterns supply motivational signals for releasing outputs only if an input exemplar from an internally desired recognition category is detected. The total circuit forms a neural network architecture which can

- stably self-organize an invariant pattern recognition code in response to a sequence of analog or binary input patterns
- be attentionally primed to ignore all but a designated category of input patterns
- automatically shift its prime as it satisfies internal criteria in response to actions based upon the recognition of a previously primed category of input patterns and
- learn to generate an arbitrary spatiotemporal output pattern in response to any input pattern exemplar of an activated recognition category.

Such circuits, and their real-time adaptive autonomous descendants, may prove useful in some of the many applications where preprogrammed rule-based systems, and systems requiring external teachers not naturally found in the applications environments, fear to tread.

Self-stabilization of speech perception and production

The insights gleaned from the design of ART 2 have also begun to clarify how we

can design hierarchical learning systems with multiple ART levels. Figure 11 shows a hierarchical ART system for learning to recognize and produce speech. The system self-stabilizes its learning in real time without using a teacher. This ART architecture is being developed at Boston University by Michael Cohen, Stephen Grossberg, and David Stork. Top-down ART expectation mechanisms at several levels of the architecture help to self-stabilize learned codes and to self-organize the selection of invariant recognition properties. Of particular interest in this speech architecture is the role of top-down expectation signals from the architecture's articulatory, or motor, system to its auditory, or perception, system. These expectations help to explain classical results from motor theory, which state that speech is perceived in terms of how it would have been produced, even during passive listening.

The key insights of the motor theory take on new meaning through the self-stabilizing properties of top-down articulatory-to-auditory expectations. These expectations self-stabilize the learned imitative associative map that transforms the perceptual codes which represent heard speech into motor codes for generating spoken speech. In so doing, the articulatory-to-auditory expectations deform the bottom-up auditory STM patterns via 2/3 Rule-like matching into activation patterns consistent with invariant properties of the motor commands. These motorically modified STM codes are then encoded in long-term memory in a bottom-up adaptive filter within the auditory system itself. This bottom-up adaptive filter activates a partially compressed speech code at the auditory system's next processing level. The motorically modified speech code is thus activated during passive listening as well as during active imitation.

Psychophysiological and neurophysiological predictions of ART

Although applications of ART to computer science depend upon the computational power of these systems for solving real-world problems, ART systems are also models of the biological processes whose analysis led to their discovery. In fact, in addition to suggesting mechanistic explanations of many interdisciplinary data about the mind and brain, the theory has also made a number of predictions

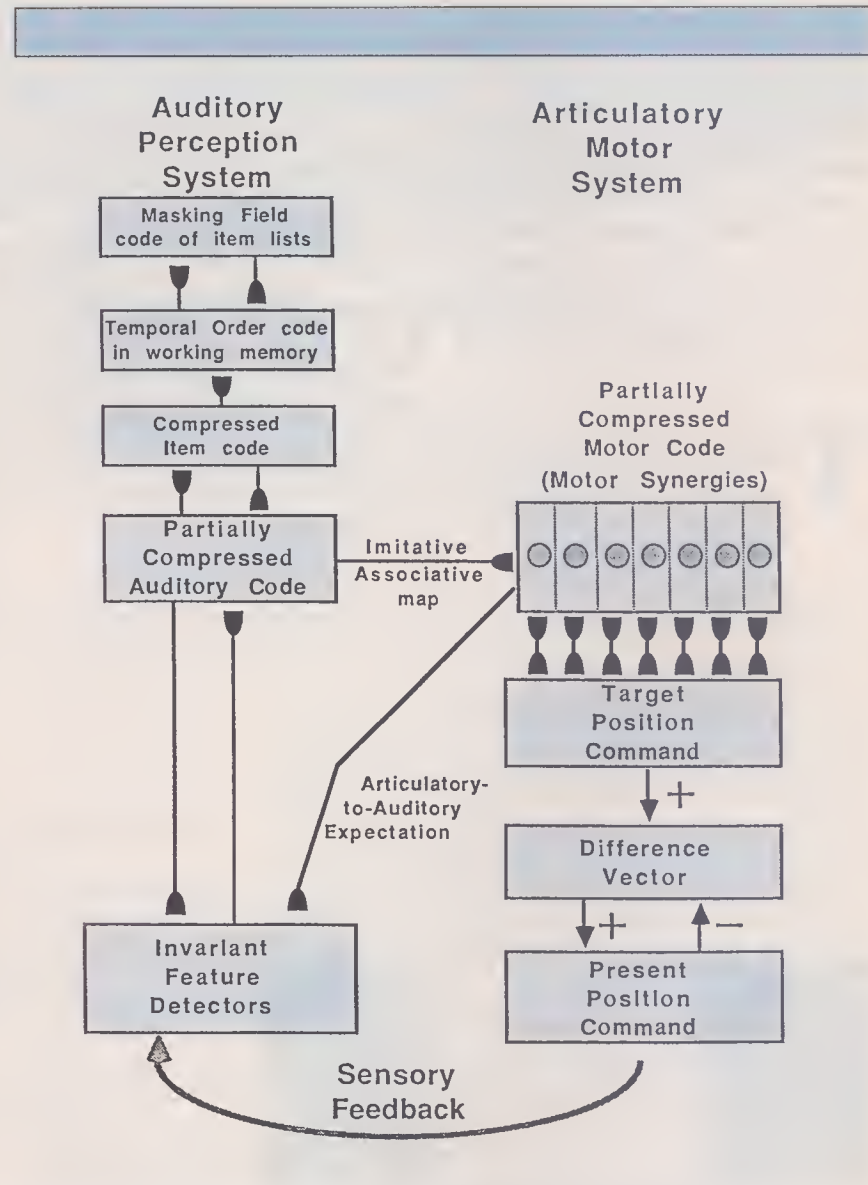


Figure 11. Schematic of some processing stages in an architecture for a self-organizing speech perception and production system. The left-hand side of the figure depicts five stages of the auditory model; the right-hand side depicts four stages of the motor model. The pathways from the partially compressed auditory code to the motor system learn an imitative associative map which joins auditory feedback patterns to the motor commands that generated them. These motor commands are compressed via bottom-up and top-down adaptive filters within the motor system into motor synergies. The synergies read out top-down learned articulatory-to-auditory expectations, which select the motorically consistent auditory data for incorporation into the learned speech codes of the auditory system.

partially supported by experiments. For example, in 1976, it was predicted that both norepinephrine (NE) mechanisms and attentional mechanisms modulate the adaptive development of thalamocortical visual feature detectors. In 1976 and 1978, Kasamatsu and Pettigrew described NE

modulation of feature detector development, and Wolf Singer reported attentional modulation in 1982. In 1978, a word length effect in word recognition paradigms was predicted. In 1982 and 1983, Samuel, van Santen, and Johnston reported a word length effect in word

superiority experiments. In 1978 and 1980, a hippocampal generator of the P300 event-related potential was predicted. In 1980, Halgren and his colleagues reported the existence of a hippocampal P300 generator in humans. The existence and correlations between other event-related potentials, such as processing negativity (PN), early positive wave (P120), and N200 were also predicted in these theoretical articles. These predictions and supportive data are described in several recent books.^{3,4}

Thus, ART systems provide a fertile ground for gaining a new understanding of biological intel-

ligence. They also suggest novel computational theories and real-time adaptive neural network architectures with promising properties for tackling some of the outstanding problems in computer science and technology today. □

Acknowledgments

We received support in part from the Air Force Office of Scientific Research, the Army Research Office, and the National Science Foundation. We also wish to thank Cynthia Suchta and Carol Yanakakis for their valuable assistance in preparing the manuscript.



Gail A. Carpenter is a professor in the Dept. of Mathematics of Northeastern University and senior research associate in the Center for Adaptive Systems at Boston University. Her current research interests are in the fields of neural networks, pattern recognition, applied mathematics, and differential equations. She is a member of the founding board of governors of the International Neural Network Society (INNS), is organization chair of the INNS first annual meeting, and serves on the editorial board of the INNS journal, *Neural Networks*.

Carpenter received a BA degree summa cum laude in mathematics from the University of Colorado in 1970, and an MA in 1972 and a PhD in 1974, both from the Mathematics Dept. of the University of Wisconsin.



Stephen Grossberg is founder and director of the Center for Adaptive Systems and a professor of mathematics, psychology, and biomedical engineering at Boston University. His current research interests include vision, cognitive information, processing, sensory-motor control, and speech perception and production. He is a member of the founding board of governors of the International Neural Network Society and currently serves as its president. He was awarded the Norbert Wiener Medal for Cybernetics and the A.P. Sloan Research Fellowship, among other honors.

Grossberg received a BA degree at Dartmouth College in 1961, an MS from Stanford University in 1964, and a PhD from Rockefeller University in 1967.

References

1. G.A. Carpenter and S. Grossberg, "A Massively Parallel Architecture for a Self-Organizing Neural Pattern Recognition Machine," *Computer Vision, Graphics, and Image Processing*, Vol. 37, 1987, pp. 54-115.
2. G.A. Carpenter and S. Grossberg, "ART 2: Self-Organization of Stable Category Recognition Codes for Analog Input Patterns," *Applied Optics*, Dec. 1, 1987, pp. 4919-30.
3. S. Grossberg, ed., *The Adaptive Brain, Vol. I and II*, Elsevier/North-Holland, Amsterdam, 1987.
4. S. Grossberg, ed., *Neural Networks and Natural Intelligence*, MIT Press, Cambridge, Mass., 1988.
5. C. von der Malsburg, "Self-Organization of Orientation Sensitive Cells in the Striate Cortex," *Kybernetik*, Vol. 14, 1973, pp. 85-100.
6. S. Grossberg, "Adaptive Pattern Classification and Universal Recoding, I: Parallel Development and Coding of Neural Feature Detectors," *Biological Cybernetics*, Vol. 23, 1976, pp. 121-134.
7. S. Amari and A. Takeuchi, "Mathematical Theory on Formation of Category Detecting Nerve Cells," *Biological Cybernetics*, Vol. 29, 1978, pp. 127-136.
8. E.L. Bienenstock, L.N. Cooper, and P.W. Munro, "Theory for the Development of Neuron Selectivity: Orientation Specificity and Binocular Interaction in Visual Cortex," *Journal of Neuroscience*, Vol. 2, 1982, pp. 32-48.
9. T. Kohonen, *Self-Organization and Associative Memory*, Springer-Verlag, New York, 1984.
10. D.H. Ackley, G.E. Hinton, and T.J. Sejnowski, "A Learning Algorithm for Boltzmann Machines," *Cognitive Science*, Vol. 9, 1985, pp. 147-169.
11. D.E. Rumelhart, G.E. Hinton, and R.J. Williams, "Learning Internal Representations by Error Propagation," D.E. Rumelhart and J.L. McClelland, eds., *Parallel Distributed Processing*, MIT Press, Cambridge, Mass., 1986.
12. J.S. Denker, ed., *Neural Networks for Computing*, American Institute of Physics, New York, 1986.
13. A.B. Gschwendtner, R.C. Harney, and R.J. Hull, "Coherent IR Radar Technology," D.K. Killinger and A. Mooradian, eds., *Optical and Laser Remote Sensing*, Springer-Verlag, New York, 1983.
14. D. Casasent and D. Psaltis, "Position, Rotation, and Scale Invariant Optical Correlations," *Applied Optics*, Vol. 15, 1976, pp. 1793-99.
15. E.L. Schwartz, "Computational Anatomy and Functional Architecture of Striate Cortex: A Spatial Mapping Approach to Perceptual Coding," *Vision Research*, Vol. 20, 1980, pp. 645-669.

Readers may write to the authors at the Center for Adaptive Systems, Boston University, 111 Cummington St., 2nd Floor, Boston, MA 02215.



IEEE INTERNATIONAL CONFERENCE ON COMPUTER-AIDED DESIGN

ICCAD-88

CALL FOR PAPERS

A CONFERENCE FOR THE
EE CAD PROFESSIONAL

NOVEMBER 7-10, 1988
SANTA CLARA, CA

sponsored by
COMPUTER SOCIETY of the IEEE
IEEE CIRCUITS AND SYSTEMS SOCIETY
in cooperation with
IEEE ELECTRON DEVICES SOCIETY
ACM SPECIAL INTEREST GROUP
ON DESIGN AUTOMATION

The 1988 INTERNATIONAL CONFERENCE ON COMPUTER-AIDED DESIGN will be held NOVEMBER 7 - 10, 1988. The Conference is oriented towards Electrical Engineering CAD professionals, concentrating on CAD for Electronic Circuit Design.

AREAS OF INTEREST

Original technical papers on (but not limited to) the following topics are invited:

- 1) **SIMULATION:** Functional, Logic, Timing, Circuit, Device and Process Simulation, Modeling.
- 2) **LAYOUT:** Placement, Routing, Floorplanning, Interactive and Symbolic Layout.
- 3) **LAYOUT VERIFICATION/EXTRACTION:** Circuit Extraction/Verification, Design Rule Checking.
- 4) **TEST:** Design for Testability, Test Pattern Generation, Fault Simulation, Built-in Test.
- 5) **DESIGN SYNTHESIS:** Register-Level and Logic Synthesis, Silicon Compilation, Formal Verification.
- 6) **DATA BASE/CAD SYSTEMS:** Expert/AI Systems, Tool Integration, Data Base Management, Hardware Acceleration, CAD Systems.

All papers should be suitable for a 25-minute presentation, and must not have been previously published.

AUTHOR INFORMATION

Authors should submit 12 COPIES of both a one-paragraph abstract and a more detailed description not to exceed 1500 words or a full-length conference paper (four pages in the Digest). Excessively long submissions will be returned to the authors.

FORMAT:

The ONE-PARAGRAPH ABSTRACT, typed on one separate page, should clearly and precisely state what is new and point out the significant results. Succinctness is required since this paragraph may be included in the Advance Program.

In the 1500 WORD OR FULL-LENGTH SUBMISSION, the author must objectively address why the proposed contribution is superior to prior work or what the significance of the contribution is, if breaking new ground. Demonstration of superiority in algorithms and strategies with heuristics is required through a description of the programming implementation and application to "real" problems. Additional mathematical proofs are welcome. The contribution should address an area of current technical interest to the CAD professional. A clear description of the new contribution, status of the work and significant examples and results should be given. References, figures and tables are not counted as part of the 1500 words.

COVER PAGE REQUIREMENTS:

Submissions should include, on the cover page: the title of the paper; the category which most closely matches the paper's content (see Areas of Interest); the full name, complete return address, telephone number and affiliation of each author; and clear identification of the individual to whom all communications should be addressed. In giving your return address, please consider that the communications for paper acceptance and mailing of the author's kit occur in the month of July.

AUTHOR'S SCHEDULE

Deadline for submissions:	April 29, 1988
Notification of acceptance:	June 30, 1988
Deadline for final version:	August 12, 1988

SEND TO:

ICCAD-88 Secretary
Electrical and Computer Engineering Department
Carnegie Mellon University
Pittsburgh, PA 15213
Telephone: (412) 268-3546.

GENERAL CHAIRPERSON

Al Jimenez
Mentor Graphics Corp.
1940 Zanker Road
San Jose, CA 95112
(408) 436-1500

PROGRAM CHAIRPERSON

Andrzej J. Strojwas
ECE Department
Carnegie Mellon University
Pittsburgh, PA 15213
(412) 268-3530

PUBLICATIONS CHAIRPERSON

Alberto Sangiovanni-Vincentelli
Department of EE and CS
University of California, Berkeley
Berkeley, CA 94720
(415) 642-4882



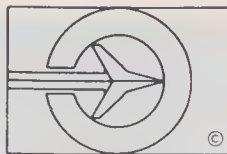
THE INSTITUTE OF ELECTRICAL
AND ELECTRONICS ENGINEERS, INC.



Association for Computing Machinery



THE COMPUTER SOCIETY
OF THE IEEE



Call for Papers

Fourth Aerospace Computer Security Applications Conference

December 12-16, 1988
Sheraton World Hotel
Orlando, Florida

Sponsored by
The Computer Society of the IEEE
The American Society for Industrial Security
The American Institute of Aeronautics and Astronautics
The Department of Defense Computer Institute
The Aerospace Computer Security Associates

The Conference

Operational requirements for civil and military systems under development increasingly stress the necessity for information to be readily accessible to users and operators. This produces an apparent conflict with policies and directives which require total protection of system data from compromises of privacy, confidentiality, and integrity. Accomplishing both of these sets of requirements requires the application of the maturing technology of computer security to new systems throughout their development cycle. In addition, operational approaches to satisfy system requirements and accommodate the implementation of engineering technology require intensified research and development.

This conference will explore technology applications in two complementary aspects: first, the policy issues and operational requirements for both civil and military systems; and second, the hardware and software tools and techniques being developed to satisfy system requirements. Special emphasis will be placed on specific examples of systems applications.

A three-day technical conference exploring the application of computer security technology will be preceded by two days of tutorials dealing with policy matters, technology applications, and other areas. Introductory and advanced surveys will be offered as well as advanced courses exploring specialized technological areas.

Application Areas

- Civil Government
- Commercial
- Aerospace
- National Security

Areas of Interest Include:

Trusted DBMSs and Operating Systems
Network Security
Current and Future Trusted System Technology
Space Station Requirements
Certification, Evaluation and Accreditation
Policy and Management Issues
Advanced Architectures
C³I Systems
Risk/Threat Assessments

Papers and Tutorials

Technical papers and tutorials which address the application of computer security technologies in the aerospace and other environments are solicited. Original research, analyses and approaches for defining the computer security issues and problems identified in the Conference's interest areas; methodological approaches for analyzing the scope and nature of computer security issues; and, potential solutions are of particular interest. Full unclassified papers, or unclassified abstracts of classified papers, must be mailed before 20 May, 1988. Material should be sent to:

Dr. William T. Bisignani
Technical Program Chairman
Booz-Allen & Hamilton Inc.
4330 East-West Highway
Bethesda, MD 20814

Tutorial Proposals including a detailed outline and a resume of presenter(s) must be mailed before 20 May, 1988 to:

Dr. Dixie B. Baker
Tutorial Program Chairwoman
The Aerospace Corporation
P.O. Box 92957
2350 East El Segundo Blvd.
El Segundo, CA 90245-4691

Classified Sessions

Classified sessions at the SECRET level are planned.

Vendor Exhibit

A technical exhibit program is planned for interested vendors displaying computer and physical security products. Inquiries should be sent to Mr. Robert D. Kovach at the address below.

Additional Information

For more information or to receive future mailings, please contact Mr. Robert D. Kovach, Publicity Chairman at: Aerospace Computer Security Associates

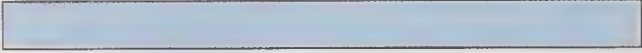
c/o ORI, Inc.
1375 Piccard Drive
Rockville, MD 20850



THE COMPUTER SOCIETY
OF THE IEEE



THE INSTITUTE OF ELECTRICAL
AND ELECTRONICS ENGINEERS, INC.



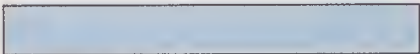
Computing with Structured Neural Networks

Jerome A. Feldman, Mark A. Fanty, and Nigel H. Goddard

University of Rochester

Rapid advances in the neurosciences and in computer science are leading to renewed interest in computational models linking animal brains and behavior. The idea of looking directly at massively parallel realizations of intelligent activity promises to be fruitful for the study of both natural and artificial computation. Much attention has been directed towards the biological implications of this interdisciplinary effort, but there are equally important relations with computational theory, hardware, and software.

Recent work on neural network computation, much of it carried out by physicists,¹ examines the emergent behavior of large, unstructured collections of computing units. We are more concerned with how one can design, realize, and analyze networks that embody the specific computational structures needed to solve hard problems. In this article, we focus on the design and use of massively parallel computational models, particularly in artificial intelligence. We also describe a computing environment for working with structured networks and present some sample applications. Throughout, we treat adaptation and learning as ways to improve structured networks, not as replacements for analysis and design.



Designing and debugging massively parallel, connectionist systems will require appropriate tools. One such package is now in use at Rochester and other laboratories.

Opportunities and limitations

Computationally, animal brains are remarkable machines with properties radically different from those of conventional computers. Our research starts from the assumption that an abstract computer based on the computational properties of animal brains may prove particularly well-

suited for problems in vision, locomotion, and language understanding. By taking seriously the computational constraints faced by nature and the structure of tasks known from artificial intelligence and other disciplines, we hope to discover algorithms employed by animals that might be effective for machines.

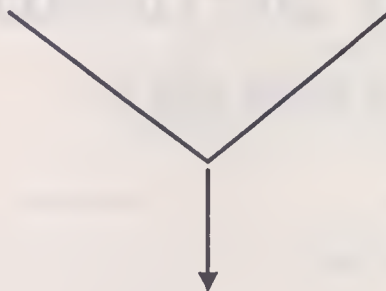
Even a crude analysis of neural computation reveals several major constraints. When asked to carry out any of a wide range of tasks, such as naming a picture or deciding if some sound is an English noun, people can respond correctly in about a half-second. The human brain, a device composed of neural elements having a basic computing speed of a few milliseconds, can solve such problems of vision and language in a few hundred milliseconds, that is, in about a hundred time steps. The best AI programs for these tasks are not nearly as general and require millions of computational time steps. This hundred-step-rule is a major constraint on any computational model of behavior. The same timing considerations show that the amount of information sent from one neuron to another is very small, a few bits at most. The range of spike frequencies is limited and the system too noisy for delicate phase encodings to be functional. This means that complex structures are not

Neural networks

Algorithms + Data Structures

relaxation
adaptation

inference
representation



Structured Connectionist Models

Figure 1. Two approaches to artificial intelligence.

transmitted directly and, if present, must be encoded in some way. Since the critical information must be captured in the connections, this is called a "connectionist" model.

The number of neurons in the human brain, estimated at about 10^{11} , presents serious constraints. For one thing, the number of cells in each retinal ganglion is greater than 10^6 , which automatically rules out any vision algorithm requiring N^2 units since there could not be a separate unit for each possible line joining two points on the retina. Many attractive algorithms for higher level tasks run afoul of the size constraint, and considerable work has gone into what amounts to coding tricks to circumvent this constraint.^{2,3}

One might think that biological constraints need not be taken seriously in designing neural-net hardware and software. Certainly, electronic switching times are a million times faster than neural ones. But the size constraint and particularly the connectivity constraint are much more serious for artificial systems. Animal brains derive much of their power from their very large fan-in and fan-out: each unit can be connected to thousands of

others. For conventional chips, a fan-out of six is quite unusual. In software simulations like the ones described in this article, excessive connectivity leads to serious performance problems. The massively parallel computational solutions evolved by nature will, no doubt, prove useful to computer science and engineering, but not without modification. Connectionist computer research aims to understand the principles of massively parallel computation and to apply them to appropriate tasks.

Connectionist models can be viewed as synthesizing two traditionally opposed approaches to artificial intelligence.⁴ (See Figure 1.) Some early AI investigators focused on the parallelism, robustness, and plasticity of animal brains and explored ways of automatically generating high performance. The other group concentrated on the detailed structure of tasks and algorithms and expressed them in conventional computer notation. Structured connectionist models attempt to capture the best of both paradigms. For example, the network fragment of Figure 2 encodes a conventional semantic network and inference structure in a parallel, evidential realization. Efforts like this one by

Shastri⁵ involve all the analysis and design issues of both AI and network theory. Recent developments in hardware and in connectionist learning theories⁶ have led some people to believe that all this might not be necessary and that uniform learning mechanisms will suffice for all the problems of interest. This is a seductive idea, but there are very good reasons—some of which are outlined below—for believing that it cannot work.

Realistic expectations. The current explosion of interest in neural networks (connectionist models, etc.) is based on a number of scientific and economic expectations, some of which are unreasonable. We can be quite sure that neural networks will not replace conventional computers, eliminate programming, or unravel the mysteries of the mind. We can expect better understanding of massively parallel computation to have an important role in practical tasks and in the behavioral and brain sciences, but only through interaction with other approaches to these problems. As always, specific structures of problems, disciplines, and computational systems are the cornerstone of success. The main hope of massively parallel (neural network) research is that it will provide a better basis for such efforts.

One particularly simplistic view of neural networks is that they will support intelligence by implementing a holographic-style memory. The basic problems with any holographic representational scheme are cross-talk, communication, invariance, and the inability to capture structure. Essentially the same problems have prevented the development of holographic computer memories or recognition systems despite considerable effort. Consider the problem of representing the concept *grandmother* as a pattern of activity of *all* the units in some memory network. Notice what happens if two (or more) concepts are presented at the same time—for example, *grandmother* at the *White House*. Obviously, if every single unit must have a specified value for the network to capture *grandmother*, then no other concept can be active at the same time. If each concept is spread over some fraction of the units, the chances are high that the encodings will overlap and cause confusion. One can reduce the probability of cross-talk by having fewer units active for each pattern. Assuming that the cross-talk is randomly distributed, Willshaw⁷ shows that the system with many concepts will be reliable (even for single entries) only if the number

ous this problem becomes. Again, any serious reduction in this wiring requirement constitutes a compact representation. And, as in the intermodal case, unless these representations are largely disjoint, concept processing must be sequential to avoid cross-talk. This eliminates the massively parallel processing required by the 100-step constraint.

Moreover, no one has suggested how to represent any but the simplest concepts in the holographic style. In fact, it is far from obvious how to make the same invariant distributed pattern active for alternative views of a chair. All proposed holographic structures are totally without internal structure. Another idea would associate the representation's components with microfeatures, but these are still unstructured. Nor do any of the holographic proposals provide a way of answering even simple questions like the color of grandmother's hair. The only general suggestion on how to encode structured knowledge in a holographic system is to encode each proposition (for example, "Dave likes candy") as a separate memory.⁸ One can build up arbitrary structures in this way, but at the cost of losing all the advantages that led to connectionist models in the first place. The possible technical advantages of exploring such models have nothing to do with human concept memory—for example, they violate the 100-step constraint by many orders of magnitude.

Another argument for highly distributed representations derives from the large number of input fibers—about 10^4 —to cortical neurons. If all these fibers participate actively, then, by its very nature, the representation is diffuse. While there has been no definitive study on the number of presynaptic events required for neural firing, estimates gleaned from papers and conversations run from one event to a few dozen.⁹ No one has suggested that several thousand synapses must fire at once for an action potential. Also, many of the connections could represent alternative ways of activating the same concept (for example, from different points in space). Another way of looking at this is that the thousand-fold connectivity is capturing an OR of activation conditions rather than an AND. Finally, learning in a connectionist system requires the potential for many more connections than are ever made functional.

All of this may seem to be flogging a dead-horse model, but purely holographic theories continue to be seriously proposed—for example, the recent flurry

of interest in spin-glasses as holographic memory models in the theoretical physics community. When addressing specific problems, Hopfield and Tank¹ use highly structured compact representations, but in general speculations they prefer holographic style models. Any physical system will, in isolation, reach some stable state, and each of these states can be looked on as encoding a different concept—obviously, a massively parallel system.

A more sophisticated version of the universal neural-net hope is that new learning techniques will induce the required structure in an initially uniform or random network. There has been significant recent progress in connectionist learning,⁶ but any nontrivial neural model also requires a great deal of prior structure. For example, the primate visual system has at least a dozen subsystems, each with elaborate internal and external connection structure.⁹ Any notion that a general learning scheme will obviate the need for neuroscience, psychophysics, perceptual psychology, and computer vision research disappears as soon as one takes the vision problem seriously, and there is no reason to believe that language, problem solving, etc., are simpler or less structured. For general computation, Judd¹⁰ has shown that even the problem of learning weights to memorize a lookup table is NP-complete and thus intractable. This implies that our formalisms, simulators, and neurocomputers must support both complex structure specification and dynamic weight change. Programming such systems, understanding their behavior, and controlling how they adapt are major continuing concerns.

The role of specialized hardware. In moving from science to technology, the issues change somewhat. Physical limits on computation speed are forcing a move to parallel systems, but not necessarily to massively parallel neural-style machines. As is well-known, current computer elements are about a million times faster than neurons, but neurons have a thousand-fold greater connectivity. These and other basic differences between electronic and living systems suggest that radically different architectures and algorithms might be appropriate in the two cases. Or maybe not; it is simply too early to tell. We do know that neural models can be effectively simulated on essentially any computer and are among the few computations that can automatically exploit an arbitrary amount of parallelism.^{11,12} For the foreseeable

future, the bulk of neural network research will (and should) be carried out with simulations on conventional computers. Building general-purpose neurocomputers is, at best, premature.

The interesting question concerns the long-term future of massively parallel systems in computation. A remarkable fact about computer architecture is its essential sameness for the vast majority of existing systems. Special-purpose architectures have had a negligible impact, and this should be a caution against neuroeuphoria. Moreover, basic reasons virtually preclude neurocomputers replacing conventional ones for most tasks. The main reason is the greatly lower cost of passive versus active elements. Many existing tasks, such as word processing, can be done adequately by a very small number of (very fast) active elements. More generally, one can view the development of writing, of mathematics, and later of computers as ways of complementing the natural capabilities of neural-style representation and computation.

If they are not going to replace conventional machines, what future is there for neurocomputers? One possibility is that calculations of physical systems will be expressed best by massively parallel networks, more or less directly simulating the situation of concern. Some low-level signal processing might best be done by parallel analog or hybrid networks.¹³ These seem quite plausible, but are a small (though important) part of computation. The best hope for widespread use of neurocomputers is, unsurprisingly, in computationally intensive areas not successfully attacked with conventional systems. The obvious ones are those that require human-like capabilities in perception, language, and inference—the traditional concerns of artificial intelligence. If such efforts succeed, there will be related applications in real-time control of complex systems and others we can't anticipate now.

If the practical future of neurocomputers depends on intelligent activity, we have come full circle—back to the scientific issues. The critical question is *do some features of massively parallel computation make it uniquely suited for calculations associated with intelligence?* The shortest path to an answer may be the indirect one of exploring connectionist models of intelligence.^{2,3} If neural-style computation has unique advantages, these should appear in the higher level specifications of perception, language understanding, and the

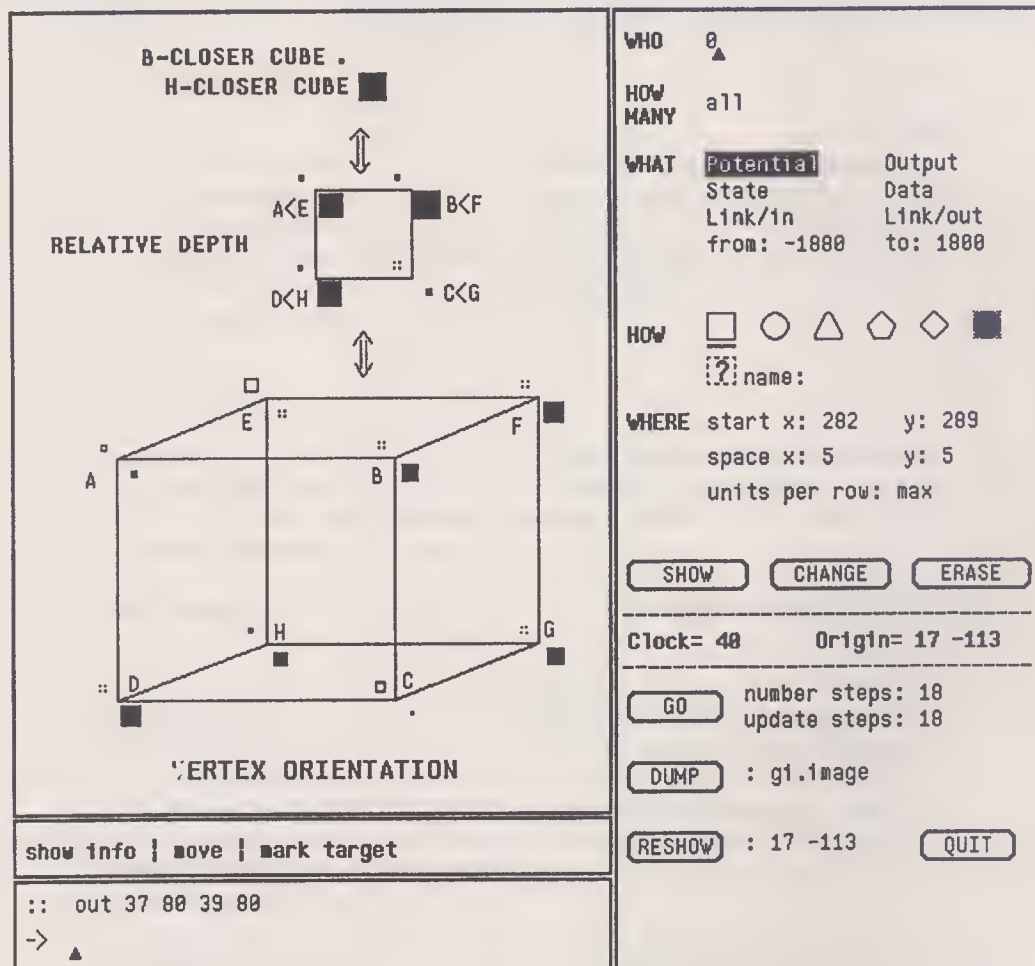


Figure 3. Networks for the Necker Cube.

like. If the conventional symbolic formalisms suffice at these levels, it is exceedingly unlikely that neurocomputers will be needed. Of course, another consequence of the convergence of scientific and technological goals is a shared interest in the detailed structure and function of natural intelligence.

When the faith in all-encompassing neural networks is abandoned, we are left with a complex set of interacting scientific and engineering issues. Rather than causing the demise of conventional computer science and engineering, neural networks present a wide range of new problems and opportunities. Thus, if we are going to design and debug neural networks, an appropriate set of tools could be very helpful. One such package, described below, has been

in use for some time at Rochester and at scores of other laboratories.

Simulation environment

We can capture some of the flavor of connectionist computation with a simple example. The cube shown in Figure 3 is a famous optical illusion originally due to the Swiss naturalist, L.A. Necker (1832). Most people initially see the cube with the vertex B closer to them, but it also can be seen as a cube with vertex H closest to the observer. If you focus on vertex H and imagine it coming out of the paper toward you, the picture will flip to the H-closer

cube. Notice also that the flip takes less than a second. The Necker cube is interesting to psychologists because it flips spontaneously between the two views if you keep looking at it. To us, it is interesting because of what it reveals about parallel computation.

You have observed how quickly the Necker cube flips state and know how slow the underlying human computing elements are. It seems unlikely that a sequential program on such a slow device could do the job. But the situation is much more complex. We know that both human and computer vision require several levels of processing. Typical levels include edge segments, lines, vertices, faces, and object descriptions. The edges and lines are the same for both the H-closer and B-closer

```

#define FRONT = 0 BACK = 1 DEPTH = 2-literal constants

CreateUnits ()
{
    for i = 0 to 25
        j = MAKEUNIT("node,"UFasympt)
        ADDSITE(j,"excite,"SFweightedsum)
        NAMEUNIT("VIEW1,"ARRAY,0,4,3)
        NAMEUNIT("B-CLOSER,"SCALAR,12,0,0)
        NAMEUNIT("VIEW2,"ARRAY,13,4,3)
        NAMEUNIT("H-CLOSER,"SCALAR,25,2,0)
}
- make units and sites
- asymptotic function
- weighted sum function
- view B units
- view B top level
- view H units
- view H top level

CreateInhibition ()
{
    for i = 0 to 7 BiLink(i,i + 13, - 575)
    for i = 8 to 11 BiLink(i,i + 13, - 800)
    BiLink(INDEX("B-CLOSER"),INDEX("H-CLOSER"), - 1000)
}
- make inhibit links
- link opposing vertex
- and depth units
- link top-level units

CreateExcite (front,back,depth,cube)
{
    for i = 0 to 3
        BiLink(front + i,front + ((i + 1)%4),200)
        BiLink(back + i,back + ((i + 1)%4),200)
        BiLink(depth + i,depth + ((i + 1)%4),300)
        BiLink(front + i,back + i,200)
        BiLink(front + i,depth + i,300)
        BiLink(back + i,depth + i,300)
        BiLink(depth + i,cube,500)
}
- make excitatory links
- for one view
- around front face
- around back face
- between depth units
- between faces
- between vertex and
- depth units
- from depth to cube
- and cube to depth

BiLink (firstunit, secondunit,weight)
{
    MAKELINK(firstunit,secondunit,"excite,"weight,NULL)
    MAKELINK(secondunit,firstunit,"excite,"weight,NULL)
}
- make symmetric links
- no link function
- link top-level units

build()
{
    CreateUnits()
    CreateInhibition()
    CreateExcite(INDEX("VIEW1,"FRONT),INDEX("VIEW1,"BACK),
    INDEX("VIEW1,"DEPTH),INDEX("B-CLOSER"))
    CreateExcite(INDEX("VIEW2,"FRONT),INDEX("VIEW2,"BACK),
    INDEX("VIEW2,"DEPTH),INDEX("H-CLOSER"))
}
- build function
- make units
- make inhibit links
- VIEW1 links
- VIEW2 links

```

Figure 4. Code for setting up the Necker cube of Figure 3. Primitives are in uppercase.

cubes, but many other visual features are seen differently in the two views. For example, vertex C is oriented into the paper in the B-closer reading, but out of the paper in the other reading. Similarly, C is closer than G in the B-closer reading, and all these perceptions are mutually con-

sistent and reinforcing. The remarkable fact is that our visual system simultaneously flips all these perceptual decisions from one mutually consistent reading of the cube to the other. This illustrates the key cooperative property of massively parallel computation and its conceptual

difference from Von Neumann computation on standard machines.

Figure 3 also illustrates some details of the connectionist paradigm. In our models, each item of interest is represented as a computational unit with connections to many other units. Each unit has a level

of activity (here between -100 and 100) and automatically sends the value of this activity along all its outgoing connections. For example, the top double arrow in Figure 3 depicts the fact that the unit representing the H-closer cube has positive two-way links with the four relative distance detectors: $A < E$, etc. Each unit at each level in this network has a rival to which it is connected by a mutual inhibition link. The strengths of links into a particular unit can also be displayed. The only other information needed for a complete model is the rule by which a unit computes its new activity from its inputs and its old activity; the simulator described in this section allows a wide variety of update algorithms. We can assume for now that the units compute the sum of their positive and negative inputs. Networks like Figures 2 and 3 are not very sensitive to the exact choice of unit computation rules; this is one of the reasons for their attractiveness. Units that are all mutually connected by negative links are said to comprise a winner-take-all network. Such networks are one of the main decision mechanisms in connectionist models and have known neurophysiological analogs.

Much of the effort in massively parallel AI is dedicated to using computational frameworks like that in Figures 2 and 3 to build models of intelligent activity. Advantages of this approach include its link to natural intelligence, increased noise resistance, and ease of implementation on parallel hardware. But the main advantage of the connectionist approach is that it provides a much better way of specifying some computations. No alternative way appears to describe the Necker cube phenomenon nearly so clearly and concisely as Figure 3.

Specifying and simulating networks. Researchers experimenting with structured connectionist networks must be able to implement and test their ideas. Since our network models presume much structure, and network architecture design is a major component in the research effort, it is necessary to develop a network description language and a simulation system that supports varied architectures. Different researchers working in diverse areas need the ability to build and simulate radically different networks. Three of the most important aspects that may differ are connection topology, activation functions, and the amount of data associated with each network node.

Connectionist networks consist of sim-

ple computational elements (units) that communicate by sending their level of activation via links to other elements. The units have a small number of states and compute simple functions of their inputs. Associated with each link is a weight, indicating the "significance" of activation arriving over that link. The pattern of connections, the weights on the links, and the unit functions determine network behavior.

The Rochester Connectionist Simulator has evolved from simple beginnings into a sophisticated research tool that supports construction and simulation of a wide variety of networks. The main design criterion has been flexibility. Each unit can compute a different function and have any amount of associated data; an arbitrary connection pattern may be specified. This flexibility exacts its cost in time and space. Compared to special-purpose simulators, it is time-expensive because each unit and link is simulated by a separate function call, and space-expensive because each unit and link must have an explicit representation.

The network paradigm supported by the simulator is quite general. Each unit has a number of sites at which incoming links arrive. The provision of sites allows differential treatment of inputs, since the links themselves do not indicate their origin at the destination site. Associated with each unit are various pieces of data, including potential, output, and state. The potential corresponds to the unit's level of activation, the output is transmitted along all links emanating from the unit, and the state is used to make simple decisions about how to interpret the unit. Associated with each link, site, and unit is a function representing its action.

Network specification. The construction environment provides a set of primitives for specifying a network. Typical primitives make a unit or link, or name a unit or array of units. The primitives provide a conceptual structure through which networks may be specified. Rather than define an entirely new language for specification, we use an existing programming language, augmented by the primitives.

A network is built in the simulator by a user program written in C. The primitives are implemented as library functions, called from the user program. The specification parameters for units, sites, and links include initial data values and an activation function. These activation functions may be different in every case, either written by the user or supplied from a

library. Within each unit, site, and link, a general-purpose data field exists, which can be used to point to an arbitrarily large structure. When displaying, saving, or reloading the network, user-supplied functions are called by the simulator to handle these user-defined structures.

The large library of functions facilitates the construction task by supplying many of the commonly used unit, site, and link functions, as well as functions to create particular network structures. Researchers may add their own unit and structuring functions to the library, thus augmenting the simulator's utility for themselves and others. An example is a library that greatly eases the construction of back-propagation networks, as either stand-alone networks or submodules within a larger network.¹⁴ This library allows arbitrary error propagation functions as well as any unit activation function.

An important consideration in specifying a network is the ability to give descriptions at different levels of abstraction. At the lowest level, the unit, site, and link functions give single-unit descriptions. At the next level, a set of functions, which specify the links and groups of units, describe the pattern of connectivity. This may include modularity within the network, and for any large network necessarily reflects regularity. At a still higher level, network specifications given in user-defined languages may be read in and compiled into units and links by user-supplied functions.⁵

A sample construction program. For an example of a network and its specification program, let's look at the Necker cube network in Figure 3. It consists of 26 units—eight pairs for the vertices, four relation pairs, and the decision units. The units corresponding to the two views are displayed as solid-line squares and dotted-line squares, respectively. The figure shows the situation when the H-closer view has almost been recognized. Most of the units consistent with this view are near maximum activation, and those of the other view at minimum.

The vertex units represent local three-dimensional orientation information. The relation units represent pairwise constraints between corresponding vertices, and the cube units represent the two possible interpretations. The links to or from a unit can be displayed with the graphics interface, which was used to generate Figure 3.

Figure 4 shows a specification program

for this network, written in pseudocode. As can be seen, the structure in the network is reflected in the structure of the specification program. Units 0 to 8 represent the B-closer cube's vertex nodes, ABCDEFGH in order. Units 8 to 11 represent the depth level, $A < E$, $B < F$, $C < G$, and $D < H$ in order. Unit 12 is the B-closer unit. Units 13 to 25 are encoded in a similar manner for the H-closer network. The linking specifications implement this choice of encoding. (See Feldman et al.¹² for details.)

Simulation. Once a network has been constructed, the simulation, run either synchronously or asynchronously, can begin. During synchronous simulation, all units use the output values computed during the previous step as their input. The order of simulation is unimportant since the network behaves as though all units update simultaneously. During asynchronous simulation, a fraction of the units are updated at each step in pseudorandom order, and the new output value is immediately transmitted to the other units. Synchronous simulation is easier to understand and is marginally faster than asynchronous, but in cases like the Necker cube example, synchronous updating with deterministic unit functions always leads to the same outcome. In other cases, synchronous simulation can lead to oscillations or other problems of failing to break symmetry.

A command interface controls the simulation and modifies the network during simulation. The interface allows execution of any user function and of a preloaded set of commands. One current trend in our simulation environment is toward a more interactive system for network specification. Dynamic code-loading capabilities allow runtime redefinition of activation functions and reconstruction of parts of the network. Integration with the Common Lisp and Scheme environments provides the added expressive power of those languages for network specification.

Performance. Two major performance issues in neural network simulation are size and performance. On a SUN3/260, a network of 2000 units (computing the weighted sum of their inputs), each with 100 links, took 25 seconds to construct and performed 100 simulation steps in 83 seconds, giving an interconnect time of approximately four microseconds. The number of links in a network is a crude but effective measure of its size; on the

SUN3/260 with eight megabytes of memory, the maximum number of links attainable before thrashing sets in is of the order of a quarter of a million. The parallel simulator, discussed below and in Feldman et al.,¹² can do much better.

While space and time efficiency of the simulation engine is crucial for experiments with large networks, clearly, ease of use, extensibility, and flexibility are equally important attributes. Future needs, such as large networks running on parallel hardware, will require fast simulation systems connected to a high-speed graphics interface.

Graphics interface. The current graphics interface, a first-generation tool developed by Kenton Lynne, has proved indispensable for displaying network information during simulation and for aiding the network debugging process. It allows the user who has created a network with the simulator to graphically view and examine that network before, during, and after execution. Each significant aspect of each unit in the network can be displayed as a separate graphic object (or icon) whose size, shape, or shading varies with the current value of that aspect. As the simulation runs, the icons are constantly updated to reflect changing values, giving an overall view of what the network is doing. In keeping with the philosophy of the simulator, the graphics interface was designed to give maximum flexibility to the user in how the network is displayed.

Lynne wrote the graphics interface specifically to run on our Sun workstations using Sun Microsystems' graphics tool package. He designed it as a separate part of the simulator package; the user specifies it as an option when the network is compiled into object code. If included, the graphics interface code automatically creates its own window, which layers itself between the user and the simulator. The user then interacts with the simulator via the graphics interface window, which either executes user-given graphics commands or passes appropriate commands to the simulator. It provides and maintains a graphics panel, which displays the graphical representation of the user's network as simulation proceeds. Figure 3 shows the graphics interface tool in potential mode examining the activations for the Necker cube example. The simulator is unaware of the graphics interface and, in fact, still generates text output to its own window, just as though the user were interfacing to it directly.

In addition to displaying and running the network, the graphics interface has facilities that allow the user to

- display detailed textual information for specific units,
- show network topology,
- write the graphics display to a raster file,
- put text and line drawings on the display for documentation purposes,
- log the simulator and graphics commands for later replay, and
- map the mouse buttons to execute simulator or graphics commands.

There is also a version that communicates with the parallel connectionist simulator running on the department's BBN Butterfly multiprocessor.

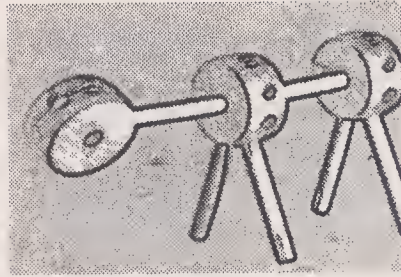
Through commands or mouse actions, the user specifies exactly where the icons are to appear in the display space, which is effectively an unbounded Cartesian plane. The graphics panel, which can be stretched vertically and horizontally, always shows some finite rectangle of the display space and can be moved via commands or mouse actions to show different parts of the display space. Thus, the size of a displayable network is limited only by machine memory. (Each displayed aspect requires 48 additional bytes of memory). The user can also add text and line drawings to the display space to document the network as shown in Figure 3. All objects (icons, text, drawings) appearing on the graphics display can be freely moved around with commands or the mouse as dictated by taste or function.

Using the graphics interface significantly reduces the time needed to get a connectionist network up and running. With the proper unit aspects displayed, one can quickly determine if a network is working properly and if not, where the problem lies. Much of the power of the interface lies in its dynamic properties, which do not show up in static pictures. For example, one can catch oscillations, which can paralyze a network, almost immediately. The graphics interface makes it much easier to specify and debug the large, structured networks we work with at Rochester.

Parallel implementation. We have also implemented the connectionist simulator on a BBN Butterfly multiprocessor. The parallel simulator looks and functions much like the uniprocessor simulator. Code can be ported with little modification, as long as it does not directly access the network data structures. If the user is content with a naive network partition, he



(a)



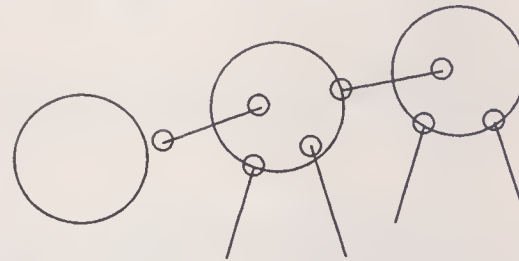
(b)



(c)



(d)



(e)

Figure 5. Tinker Toy low-level processing: (a) original figure; (b) response to Kirsch operator; (c) extracted lines; (d) extracted circles; (e) final symbolic representation, including connections detected between parts.

can ignore the fact that the simulator is actually running on several processors.

Using the Butterfly multiprocessor significantly increases the speed and capacity of network simulations. Our largest Butterfly configuration is 120 processors with a total of 120 megabytes of memory. It easily runs networks that do not begin to fit on any of our uniprocessor machines and achieves nearly linear speedup.¹² Even with smaller networks, simulations that would run for hours on a uniprocessor take only minutes.

Sample applications

The driving force behind the system developments described above has been applications of connectionist models, particularly to problems in artificial intelligence. Since all connectionist networks are currently simulated on inappropriate hardware, applications refer to concept demonstrations and scientific models, not to programs for practical use. The literature^{2,3} and this issue of *Computer*

describe many such applications. We will focus on some recent Rochester work that illustrates the structured connectionist style and the use of the simulator.

The first example is a rather ambitious vision project carried out by Paul Cooper and Susan Hollbach¹⁵ with help from Steven Whitehead and others. The basic problem is to match real images of Tinker Toy objects to stored topological models. This is a simplified model of how a neural style network could perform visual recognition in the required 100 time steps. In previous research, Cooper and Hollbach established that the early stages of vision can be done effectively in parallel; the problem is how to do high-level matching to object models. The low-level vision preprocessing for this is depicted in Figure 5. Figure 5a shows a digitized picture of a Tinker Toy horse, the input to the system. Figure 5b represents the magnitude of the gradient produced by the Kirsch operators (that is, the edge picture.) Figures 5c and 5d have the straight lines and circles found by the Hough technique from the edge

information of 5b. Figure 5e demonstrates the connectivities found between rods and disks: the small circles represent joints between a rod and disk.

Figure 6 gives a typical model database. The idea is to match the analyzed input to all the models in parallel using a structured connectionist network. Figure 7 shows a model horse, labeled by letters, and a possible input image, labeled by numbers. The matching network tries to match simultaneously compatible wheels (ones with the same number of rods) and to ensure that adjacent wheels in the image match adjacent model wheels. These mutual constraints are adequate to select the correct model for a wide range of tasks. Figure 8 shows some of the consistency constraints; for example, if the image pair 3-4 matches the model pair B-C, then 3 must match B or C. The details of the model are moderately complex, involving several winner-take-all networks and simultaneous comparison of all models.

Figure 6 also depicts the results of matching the results of Figure 5 with the

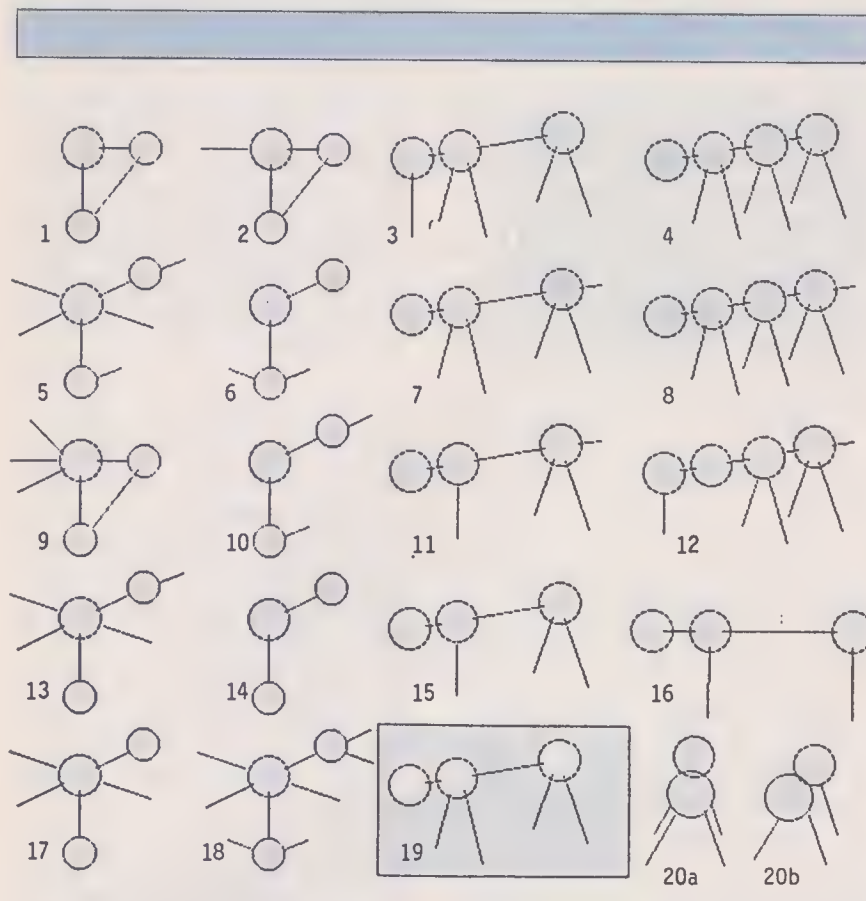


Figure 6. Tinker Toy model base.

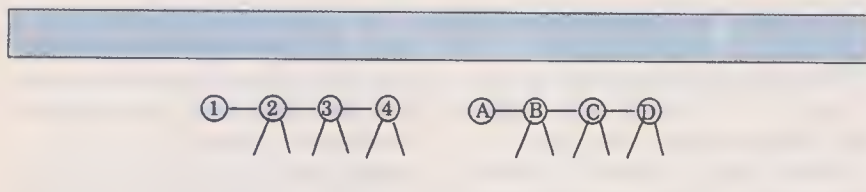


Figure 7. Example of a figure and matching model.

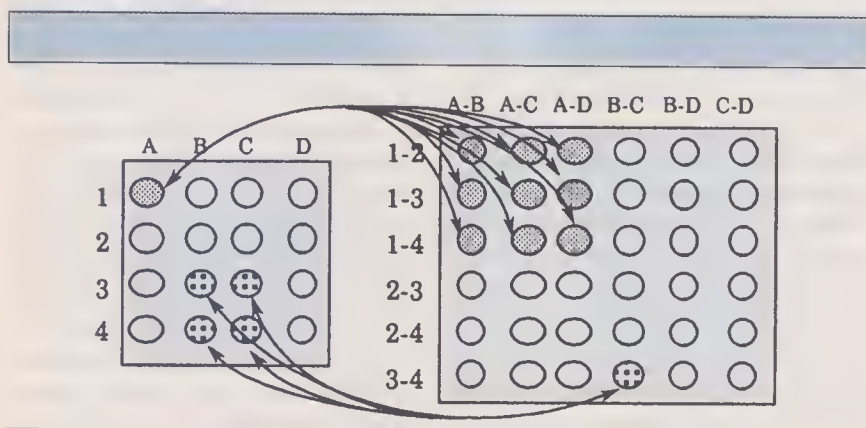


Figure 8. Wheel-matching array, constraint-matching array, and example constraint propagation links.

21 models. The central goal of recognizing the target horse figure is accomplished, and reasonable scores are obtained by partial matching of approximately similar figures. Although the network was designed primarily to accept or reject matches of objects with the same number of disks, it also computes partial matches and matches of objects with differing numbers of disks. For example, compare the extracted figure with models 3 and 7 (judged to be topologically quite similar) and models 4, 6, and 15 (judged fairly similar).

The Necker cube and Tinker Toy examples are instances of AI recognition problems. Several other problems are like this, but many others are not. Can we apply structured connectionist models to other traditional AI issues such as knowledge representation and inference? There is much less completed research along these lines, but some promising starts have been made. The example in Figure 2 should convey the flavor of this work.

The standard way to explore the issue of knowledge representation and inference is in terms of programs that can answer questions. The many AI approaches to developing question-answering systems have the same basic requirements: One needs a way to store knowledge, to pose questions, and to compute and register answers. In a connectionist model, all these aspects must be expressed in terms of activity spreading among simple units like those in the previous examples.

It is easiest to start with the recording of answers. In Figure 2, the possible tastes of foods form a winner-take-all network in which each unit inhibits the others so that only one answer will be active. The answer network is assumed to be part of a routine that also poses the question and acts upon the answer. The units that make up the routine are assumed to be activated in sequence from left to right, just like a standard program. Activating the appropriate units sends a question to the knowledge network; Figure 2 shows this as links from the hexagonal node to the nodes for [has-taste] and [ham]. The key to this network's operation is operation of the triangular-shaped nodes such as [b1] in Figure 2. A unit shown as a triangle is defined to become active when two of its inputs are simultaneously active. In this case, [ham] and [has-taste] are both on, so [b1] becomes active and activates [salty]. Now the [salty] node in the knowledge network spreads activation to the response [r-salty] back in the routine and the question is

answered. The same network can answer questions like "Name a salty meat" when activated appropriately. The answers returned by such a network depend on context, as people's answers do; contextual bias is again modeled by activation.

In addition, Shastri⁵ showed that structured connectionist knowledge representations can handle problems that have proven difficult for logic-based approaches. Suppose we believe that Quakers tend to be pacifists and that Republicans are generally not pacifist. Given an individual who is both a Quaker and Republican, it is hard to decide how likely he is to be a pacifist. (The recent US President who had these two beliefs was also a Marine officer). Shastri's system allows the relative strengths of conflicting beliefs and correlations to be combined according to maximum entropy rules of evidence and performs quite well. Again, the structured connectionist network provides a natural mechanism for representing the required knowledge and for capturing inferences based on partial, uncertain, and conflicting knowledge.

An obvious question arising in connection with work like Shastri's is how a network like Figure 2 could be learned. The neural substrate of memory and learning is one of the great unsolved scientific questions for which we certainly have no definitive answers. But there are connectionist theories of learning that are compatible with current brain research and are computationally feasible.⁶ The key idea is that while new connections are rare, *weight change* in connections appears to be common. We also know that each unit can have thousands of incoming and outgoing connections. Our hypothesis is that most of these connections are only potentially important and that learning involves strengthening the appropriate connections. Suppose, for example, the network of Figure 2 needs to learn that spinach is a salty vegetable. Our model suggests that there are uncommitted triangular nodes that are weakly connected to many combinations of objects, properties, and values. In an ideal case, one of them will be linked to [spinach], [has-taste] and [salty] among other things. This unit will get highly activated by the simultaneous activation of three of its neighbors and, by strengthening its active connections, can become dedicated to the new association. No implementation of this learning scheme has been attempted, but there are several related studies and some theoretical work.¹⁶

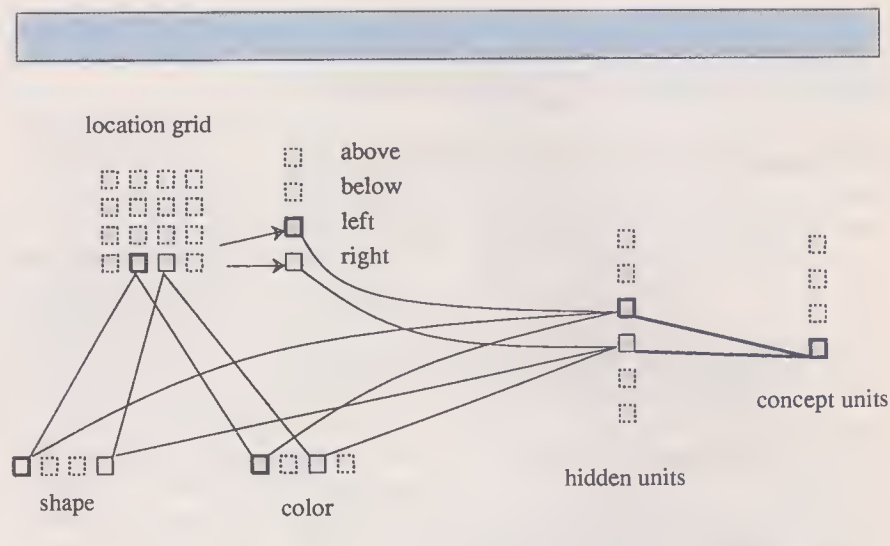


Figure 9. Input consisting of two objects, with attention focused on the leftmost of the pair.

Fanty's work¹⁷ is our most ambitious effort to date on learning in structured neural networks. An example system based on his work learns to classify structured objects in a position-independent manner. Both structured objects and position independence present difficulties to the connectionist researcher for essentially the same reason. If a network is going to perform computations at several locations, or for several subparts of a complicated object, at the same time, then there must be several copies of the computational mechanism. This is problematical. It can lead to a combinatorial increase in the required number of units and links, an especially troublesome problem in learning networks where something learned locally should be shared globally. The proposed solution is to go sequential. At some point, even connectionist networks must begin to compute sequentially. Imagine a story understander for which the input is entered simultaneously: all 100,000 words of a novel, each in its own word slot. The whole network is replete with redundant sentence-understanding subnetworks, subplot-understanding subnetworks, and so on. This, of course, is ridiculous. The idea behind the network described here is that some sequential processing is always necessary in reasoning about multiple objects; there can, however, be resource-intensive exceptions, such as low-level vision.

For the implemented networks, each input consists of some number of subob-

jects, each with a shape and color. The subobjects are located on a 4×4 grid. So a single input might be a red square over a blue circle, and it could appear in any of 12 different locations. For each input, the network is provided feedback on classification. The goal is to have the network learn to classify the inputs.

Figure 9 illustrates the network's organization. A grid of units corresponding to location is the extent to which the representation is spread in space. There are four other populations of units, each with a distinct role. There is a network of object properties, shown with four values for shape and color. There is a network of relations, shown with the four spatial relations: above, below, left, and right. There is a population of hidden units, which will learn to classify subobjects. Finally, there is some number of classification units. Each complex input belongs to one of the classes. The network learns to classify the inputs based on feedback at this level.

Notice that the position independence of the learning is guaranteed; the network is prestructured, which is the whole point. Learning occurs at the hidden and classification units, which receive input from the property and relation units. The activity of these latter units is independent of the input's location on the grid. The nature of the representation needs more detailed explanation. Each grid unit corresponds to a location in the world. When one of these is active, it means something at that location is being attended to. The

grid units inhibit each other so that at most one can be strongly active. The grid units representing other objects in the current input are not totally quiescent. They maintain a low level of activity, not enough to interfere with the objects of attention, but enough to keep them at the ready (and, potentially, enough to prime ongoing computations and shifts of attention). After being active for a while, a grid unit will tire and fall back to a low level of activity, and the strongest of the waiting grid units will reach full activation. External influences could control the shift of attention through priming effects, though this does not happen in the example network.

An object is entered one subpart at a time by externally activating a grid location and some properties. The external activation is very strong and causes a temporary binding between the grid object and the properties. This is effected via a large change in the link weight from the grid unit to the property unit. The weight change will stay in effect until the grid unit becomes totally quiescent. So, as the grid units cycle up and down, the associated properties do as well. In this example, the relations are all spatial and are activated by hard-wired relation detectors (not shown). In Figure 9, grid unit (4,2) is strongly active and grid unit (4,3) is weakly active. This activates the left relation: the focused-upon object is left of something. It also has shape 1 and color 1.

The job of the hidden units is to classify the subobjects. Hidden units are only responsive to currently active units. In Figure 9, an active hidden unit responds to the triple: (shape = 1, color = 1, left). Another hidden unit responds to (shape = 4, color = 3, right), but it is shown at a low level of activation because that object is not being attended to. In the current implementation, the learning algorithm used for this stage is a variation of competitive learning.¹⁴ This was chosen because it does not require feedback, which is harder to provide in such a dynamic environment. However, lack of feedback has numerous disadvantages, and research into other methods is ongoing. At the classification-unit level, evidence for complex objects must be accumulated over time. In this implementation the links to classification units have a memory. Thus, if they were recently active, they continue to provide support even after the source of the activation has died down. This is not the ultimate solution, but it does work in this simple example. The learning rule for these

links is simple. Active inputs to an active classification are increased; others are decreased.

Many extensions and improvements are possible. The grid units could be replaced by general-purpose control units with no inherent semantics, which bind to object properties and relations much as the grid units do. Another idea, treating relations as distinct from properties at the hidden-unit level, would allow a distinct subobject learned in one context (for example, "below") to be used in another context. Of course, the subobject distinction is artificial since a complete object in one situation can be a subobject in another. Finding clean ways for connectionist systems to switch levels of elaboration is another topic of current research in several laboratories.

As these examples help point out, massively parallel, neural network-style computation presents a wide range of opportunities and challenges. It is certainly not a magic

universal solution, but it does offer potentially important benefits in speed, robustness, and adaptability. To take advantage of these opportunities, we must merge neural-net techniques with those of conventional computing and other sciences. Our methodology and tools are presented as a step in that direction. □

Acknowledgments

Many thanks are due Liudvikas Bukys for his assistance with the parallel implementation of the connectionist simulator and Susan Hollbach, who helped extensively with this article.

This work was supported in part by NSF/CER Grant No. DCR-8320136, in part by ONR/DARPA Research Contract No. N00014-82-K-0193, and in part by ONR Research Contract No. N00014-84-K-0655.

The Xerox Corporation University Grants Program provided equipment used in the preparation of this article.

References

1. J.J. Hopfield and D.W. Tank, "Computing with Neural Circuits: A Model," *Science*, Aug. 8, 1986.
2. D. Waltz and J.A. Feldman, Eds., *Connectionist Models and Their Implications*, Ablex Publishing Corp., 1987.
3. J.L. McClelland and D.E. Rumelhart, Eds., *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Vol 2: Applications*, MIT Press/Bradford Books, 1986.
4. A. Newell, "Intellectual Issues in the History of Artificial Intelligence," in *The Study of Information: Interdisciplinary Messages*, F. Machlup and U. Mansfield, eds., John Wiley and Sons, New York, 1983.
5. L. Shastri, *Semantic Nets: Evidential Formalization and Its Connectionist Realization*, Morgan-Kaufman, Los Altos/Pitman Publishing, London, Feb. 1988.
6. G.E. Hinton, "Connectionist Learning Procedures," *AI Journal*, in press.
7. D. Willshaw, "Holography, Associative Memory, and Inductive Generalization," in *Parallel Models of Associative Memory*, G.E. Hinton and J.A. Anderson, eds., Lawrence Erlbaum Assoc., Hillsdale, N.J., 1981.
8. D.S. Touretzky, and G.E. Hinton, "Symbols Among Neurons: Details of a Connectionist Inference Architecture," *Proc. IJCAI*, Los Angeles, 1985, pp. 238-243.
9. G. Shepard, in *The Neurobiology of Learning and Memory*, G. Lynch, ed., MIT Press, Cambridge, Mass., 1986.
10. J.S. Judd, "Complexity of Connectionist Learning with Various Node Functions," *Proc. IEEE Int'l. Conf. on Neural Networks*, San Diego, June 1987.
11. G. Brelloch, and C.R. Rosenberg, "Network Learning on the Connection Machine," in *Connectionist Models and Their Implications*, D. Waltz and J. Feldman, Eds., Ablex Publishing Corp., 1987.
12. J.A. Feldman et al., "Computing with Structured Connectionist Networks," *Comm. ACM*, Feb. 1988.
13. J. Hutchinson et al., "Computing Motion Using Analog and Binary Resistive Networks," *Computer*, Mar. 1988.
14. D.E. Rumelhart and J.L. McClelland, eds., *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Vol 1: Foundations*, MIT Press/Bradford Books, 1986.
15. P.R. Cooper and S.C. Hollbach, "Parallel Recognition of Objects Comprised of Pure Structure," *Proc. DARPA Image Understanding Workshop*, Los Angeles, Feb. 1987.
16. J.A. Feldman, "Dynamic Connections in Neural Networks," *Biological Cybernetics*, Vol. 46, 1982, pp. 27-39.
17. M.A. Fanty, "Learning in Structured Connectionist Networks," forthcoming doctoral dissertation, Computer Science Dept., University of Rochester, 1988.



Jerome A. Feldman is a professor of computer science at the University of Rochester. He was recruited in 1974 to chair the new department of computer science, and in 1981 was named the John H. Dessauer Professor of Computer Science. His research interests center around massively parallel (connectionist) computation and its applications to complex problems such as those of artificial intelligence.

Feldman received his PhD in math and computer science from Carnegie Mellon University in 1964.



Mark A. Fenty is now a research associate at the International Computer Science Institute in Berkeley, Calif. His work focuses on connectionist models of high-level cognition and learning.

Fenty was a PhD student at the University of Rochester when this article was written.



Nigel H. Goddard is currently on the technical staff of the Hughes Aircraft Company's Artificial Intelligence Center and is a research consultant to the Division of Biology at the California Institute of Technology. His current research focuses on recognition from visual motion, representation of time, connectionist simulation systems, and the bearing that neuroscience has on these three subjects.

Goddard was a graduate student at the University of Rochester when this article was written. He is a member of the International Neural Network Society.

Readers may write to Feldman at the Dept. of Computer Science, Rm. 734, Computer Studies Bldg., University of Rochester, Rochester, NY 14627.

If you want to know about the quality of opportunities at AMD today, just take a look at our RISC Microprocessor Program.

The fact is, AMD has a RISC program that's setting the standard for the next generation of 32-bit CMOS microprocessors. The Am29000 is the most powerful monolithic microprocessor ever, capable of sustained performance of 17 million instructions per second (MIPS), with a peak of 25 MIPS. Already it goes beyond RISC concepts; each facet of its architecture, from its RISC-like instruction set to its system interface, is streamlined for speed. And it promises — as do the products that will follow it — to open the door to new systems possibilities far beyond the performance of today's architectures.

If you're interested in a key role in this exciting program, and possess a BS/MS in EE or CS, here are positions currently available in Silicon Valley and in Austin, Texas:

Software Engineers

Support the compilers, assemblers, utilities, and software tools needed in a microprocessor development environment. Supervisory experience and knowledge of UNIX* and MS DOS are essential.

Senior Strategic Development Engineers

If you know RISC architecture, can program in C and assembly language, are familiar with development software, and have experience implementing IEEE floating-point arithmetic, we want to talk with you.

Senior Product Planning Engineers

We need people with a solid background in microprocessor and systems architecture design to support chip design, marketing, and tools development.

Interested? If you feel your background qualifies you for any of these positions, then let us know right away. Call Mike Gore at:

(408) 749-5555

If you are unable to call during business hours, please forward your resume to Advanced Micro Devices, 901 Thompson Place, Dept. 0204, P.O. Box 3453, Sunnyvale, CA 94088. We are an equal opportunity employer.

*Unix is a trademark of AT&T Bell Labs.



One great company



Advance Notice for the 15th Annual International Symposium on Computer Architecture



May 30th - June 3rd, 1988 Ilikai Hotel, Honolulu, Hawaii, USA

Sponsored by

Computer Society of the Institute of Electrical and Electronics Engineers, TCCA
Association for Computing Machinery, SIGARCH

General Chair: H. J. Siegel, Supercomputing Research Center
Program Co-Chairs: Doug DeGroot, Texas Instruments
Yale N. Patt, UC, Berkeley

Symposium Committee:

Finance, Local Arrangements: Mary Jane Irwin, Pennsylvania State Univ.
Tutorials: Shreekant S. Thakkar, Sequent Computer Systems
Student Grants: Mark A. Franklin, Washington Univ.
Registration: Nathaniel J. Davis IV, Air Force Institute of Technology
Publicity: Steven P. Levitan, Univ. of Pittsburgh
Proceedings: Jose Fortes, Purdue Univ.
Exhibits: Alan D. Berenbaum, AT&T Information Systems
Audio Visuals: S. Diane Smith, Tesuji

Program Committee:

Jean-Loup Baer, Univ. of Washington
Al Despain, UC, Berkeley
Jack Dongarra, Argonne Nat'l Labs

Mike Flynn, Stanford Univ.
Gideon Frieder, Syracuse Univ.
Dan Gajski, UC, Irvine
Lee Giles, AFOSR
Jim Goodman, Univ. of Wisconsin
Bert Halstead, MIT
Kai Hwang, Univ. of Southern California
Wen-mei Hwu, Univ. of Illinois
Joe Linn, Institute for Defense Analyses
Jack Lipovski, Univ. of Texas
Trevor Mudge, Univ. of Michigan
Janak Patel, Univ. of Illinois
Andy Pleszkun, Univ. of Wisconsin
Constantine Polychronopoulos, Univ. of Illinois
Dan Siewiorek, Carnegie-Mellon Univ.
Dick Sites, DEC
Jack Stankovic, Univ. of Massachusetts

Papers presented will be selected from among the record number of over 250 submissions
Tentative Conference Schedule:

Monday, May 30 Morning: (half day tutorials)

- Functional Programming and Computer Architecture
Peter Harrison and Tony Field, Imperial College, London
- Parallel Processing Systems
H. J. Siegel, Supercomputing Research Center
- GaAs Microprocessors: Architecture and Design
Veljko Milutinovic, Purdue Univ.

Monday, May 30 Afternoon: (half day tutorials)

- Logic Programming Architecture
John Conery, Univ. of Oregon
- Fine Grained Concurrent Computing
William Dally and Andrew Chien, MIT
- Cache Design for Multiprocessor Systems
Shreekant S. Thakkar, Sequent Computer Systems

Tuesday, May 31:

- All Day Technical Sessions
- Conference Cocktail Party

Wednesday, June 1:

- All Day Technical Sessions
- Conference Banquet: Presentation of the Eckert-Mauchley Award (planned)

Thursday, June 2:

- All Day Technical Sessions
- Luau (optional)

Friday, June 3: (full day tutorial)

- Neural Networks
Cliff Penn, Texas Instruments

Notes: (1) We reserve the right to cancel the tutorials if lacking significant advance registration. (2) Early registration for tutorials is encouraged due to limited seating. (3) Scholarships for the tutorials are available for students attending them at their own expense. Contact S. Thakkar (503) 627-9822. (4) The cocktail party and banquet are included in the registration. (5) The annual joint SIGARCH/TCCA meeting will be held at the symposium. (6) Spouse's activities planned.

Accommodations: A block of rooms has been reserved (until May 1st) at the Ilikai Hotel (800) 367-8434. These rooms are available at a discounted price to people who identify themselves as Computer Architecture Symposium participants. Reduced rates are for the dates of May 20th to June 12th. The number of rooms is limited and after May 1st the block will be released. Please register early. **Travel:** The official symposium travel agency is Irvine Travel Service, West Lafayette, IN. Call toll free: (800) 227-7477 or: (317) 743-2116, for information about special car rental rates and discounted air fares (5% off lowest published air fare).

For information about becoming an exhibitor call Alan Berenbaum (201) 582-3655

For Registration Forms and the Complete Advance Program

see the April issue of Computer Magazine

or call (212) 869-7440 or write to ACM Conferences Division
11 West 42nd St. New York City, NY 10036 for advance information

*** Advance Registration Ends May 8th ***



Self-Organization in a Perceptual Network

Ralph Linsker
IBM Research

A young animal or child perceives and identifies features in its environment in an apparently effortless way. No presently known algorithms even approach this flexible, general-purpose perceptual capability. Discovering the principles that may underlie perceptual processing is important both for neuroscience and for the development of synthetic perceptual systems.

Two important aspects of the mystery of perception are

- (1) What processing functions does the neural "machinery" perform on perceptual input, and what is the circuitry that implements these functions?
- (2) How does this "machinery" come to be?

Unlike conventional computer hardware, neural circuitry is not hard-wired or specified as an explicit set of point-to-point connections. Instead it develops under the influence of a genetic specification and epigenetic factors, such as electrical activity, both before and after birth. How this happens is in large part unknown.

Biological development processes are far too complex to hope that a relatively complete understanding of how a perceptual system develops and functions will soon emerge. But we are familiar with complex synthetic systems, such as computers, whose principles of organization can be understood without one's knowing

How can a perceptual system develop to recognize specific features of its environment, without being told which features it should analyze, or even whether its identifications are correct?

in detail how the components work. Furthermore, the same principles can be used to build computers in any of several different technologies. Might there be organizing principles

- (1) that explain some essential aspects of how a perceptual system develops and functions;
- (2) that we can attempt to infer without waiting for far more detailed experimental information; and

- (3) that can lead to profitable experimental programs, testable predictions, and applications to synthetic perception as well as neuroscientific understanding?

I believe the answer is yes, and that the use of theoretical neural networks that embody biologically-motivated rules and constraints is a powerful tool in this study.

This optimism is encouraged by recent work¹ in which I have found that a multilayered network, developing according to simple yet biologically plausible "Hebb-type" rules,² self-organizes to produce feature-analyzing "cells." These "cells" have response properties that are qualitatively similar to those cells of the first few processing stages of the mammalian visual system.³ These properties include sensitivity to light-dark contrast and sensitivity to the orientation of an edge or bar. These properties develop before birth in certain animals, hence before structured visual experience, and in the theoretical network the corresponding properties develop even in the absence of structured input, using only random signaling activity in the input layer of the network.

Why does a feature-analyzing function emerge from these development rules? Is it a mere accident or curiosity? Or are the development rules perhaps acting to optimize some quantity that is important to the information processing function of a perceptual system?

In this article, I briefly summarize the network ideas from an earlier publication¹ and review some of the main results. This sets the stage for exploring why a feature-analyzing function emerges. I then show that even a single developing cell of a layered network exhibits a remarkable set of optimization properties. These properties are closely related to issues in statistics, theoretical physics, adaptive signal processing, the formation of knowledge representations in artificial intelligence, and information theory.

Next, I use these results to infer an information-theoretic principle that can be applied to the network as a whole, rather than a single cell. The organizing principle I propose is that the network connections develop in such a way as to maximize the amount of information that is preserved when signals are transformed at each processing stage, subject to certain constraints.

I illustrate how this principle works for some very simple cases. Much more work will be needed to apply the principle to practical computations of biologically important cases, but the approach appears very promising. I conclude with some speculative comments on why this principle, or some variant of it, may be important for the emergence of perceptual function in biological and synthetic systems.

A layered self-adaptive network

The visual system is the best studied perceptual system in mammals. Visual information is processed in stages. Simple aspects of form, such as contrast and edge orientation, are analyzed in the earlier stages; more complex features are analyzed later. Other aspects of visual processing, such as color and motion analysis, proceed in parallel with the analysis of form.

Both the retina and cortex are organized into layers of cells with interconnections within and between layers. Within an anatomical layer, at least for the early processing stages, there is a population of cells each of which performs approximately the same processing function on its inputs. This population of cells can be thought of as an array of filters. Each cell processes input from a limited region of visual space, called the "receptive field" of that cell. More than one population of cells can share an anatomical layer.

Many cells respond to input activity by firing an electrical pulse, or *action potential*, that travels down the output fiber, or axon. These pulses cause a chemical neurotransmitter substance to be released at synapses, or regions of near-contact with other cells. The latter cells receive and process these chemical input signals. Some cells, for example in the retina, do not produce action potentials, but instead exhibit more graded electrochemical phenomena that can be used for signaling.

Although a cell's response function is in general nonlinear, visual neurophysiologists have found that for many cells, a linear summation approximation is appropriate. In this approximation, the cell's output response varies monotonically with some linear combination of the cell's input signal values. For cells that produce action potentials, the output response can be defined as the firing rate at which the cell generates action potential pulses in response to its input signals.

Specification of the network. Will a simple self-adaptive network develop feature-analyzing cells without our specifying which features are to be analyzed? If it does, are these cell types related to those observed in biological systems? To address these questions, we first study a network that embodies some of the important biological properties described above, but omits many complicating factors. This approach is useful both because many of the details are unknown, and because our goal is to understand what principles are most important for the development of perceptual functions. For example, if we want to know how nonlinearity of response may be important for development, it is valuable to see first whether a linear response system exhibits the main feature-analyzing properties that are biologically observed. Also, feedback connections from later to earlier processing stages are known to exist, but it is not known how these connections might relate to the development of feature-analyzing functions. (There are many other functions that feedback may serve, such as control of dynamic range, attentional mechanisms, and so on.) We choose to analyze networks without feedback, to understand their developmental properties first.

The interconnections within the retina are known to be more complicated than a simple feedforward arrangement. Also, mechanisms that are not dependent on neural activity appear to be involved in the development of some feature-analyzing

properties. The main purpose of our simulations is to explore what types of simple yet biologically plausible development rules *suffice* to generate feature-analyzing cell assemblies, rather than to rule out other ways of generating them. From the results of our simple model, we will infer a potential organizing principle that can encompass nonlinear cell response, more complex connectivity, and a variety of ways of forming and modifying connections.

Our network is shown in Figure 1. The cells are organized into two-dimensional layers A, B, C, and so on, with feedforward connections to each cell from an overlying neighborhood of cells of the previous layer. Layer A receives input from the visual world (if there is any such input). We focus especially on the case in which there is no input, but instead only random activity of the cells of layer A, with no correlation of activity from one cell to the next. This activity resembles random noise or snow on a TV screen. We consider this case in order to understand how certain feature-analyzing cells may emerge even before birth, as has been observed in certain primates.

The positions of the connections to each cell need not be regular as in Figure 1, but can be chosen randomly according to a density distribution, such as a Gaussian, that favors connections from nearby cells of the previous layer. For simplicity, these positions are fixed for the duration of the development process. Each cell, at each time, has some signaling activity which we denote by a real number. Each cell exhibits a simple linear response, that is, the output is a linear combination of the inputs, with each input being weighted by a *connection strength* that will develop in a certain way. Each model cell thus acts as a linear filter.

Two points should be noted:

(1) Defining the output response as a nonlinear, for example sigmoid, function of the weighted sum of the inputs would more closely approximate some properties of the firing rates of biological neurons. These are always nonnegative and saturate at some maximum rate. However, we will see that even a linear response rule can lead to the formation of feature-analyzing cells, and we will explore what properties of linear adaptive filters are responsible for this formation. Some of the insights gained will be applicable to the nonlinear response case as well.

(2) Any transformation implemented by a feedforward sequence of layers of lin-

ear filters is a linear transformation, and hence could be implemented by a single layer of connections with properly chosen, or in this case hardwired, connection strengths. However, our purpose is not to implement a particular transformation, but rather to study what transformations are learned by a network without supervision. This multistage learning process depends upon the presence of multiple network layers.

A Hebb rule. For the development process, we use a version of an idea proposed by the neuropsychologist Donald Hebb in 1949. This idea has been central to much work on synthetic neural networks over the years, as well as to the thinking of neuroscientists about how the development of synaptic connections may relate to memory and learning phenomena. Hebb's idea was that if cell 1 is one of the cells providing input to cell 2, and if cell 1's activity tends to be "high" whenever cell 2's activity is "high", then the future contribution that the firing of cell 1 makes to the firing of cell 2 should increase.

In the language of neural networks, the connection strength is increased, or made more positive. A mathematical formulation needs to be more precise than this, and state under what conditions the strength may decrease. We use a form in which the change in strength contains a term proportional to the product of input and output activities at that connection. The Hebbian idea of modifying connection strengths according to the degree of correlated activity between input and output is central to what follows.

For an analogy to a Hebbian rule, consider a group of people whose collective opinion on a question is by definition the weighted average of the opinions of its members. If, over time, a member's opinion tends to agree with the group's opinion, then the analog of the Hebb rule states that the individual member's vote on future issues is to be weighted more strongly. The member's vote is given less weight, or even negative weight, if he consistently disagrees with the group's opinion. This type of positive-feedback control of weighting factors tends to lead to consensus within the group. As we shall see, it has other surprising consequences for the properties of the group, or output cell, response.

Mathematical formulation. This subsection and the next summarize simulations that are described in detail in my

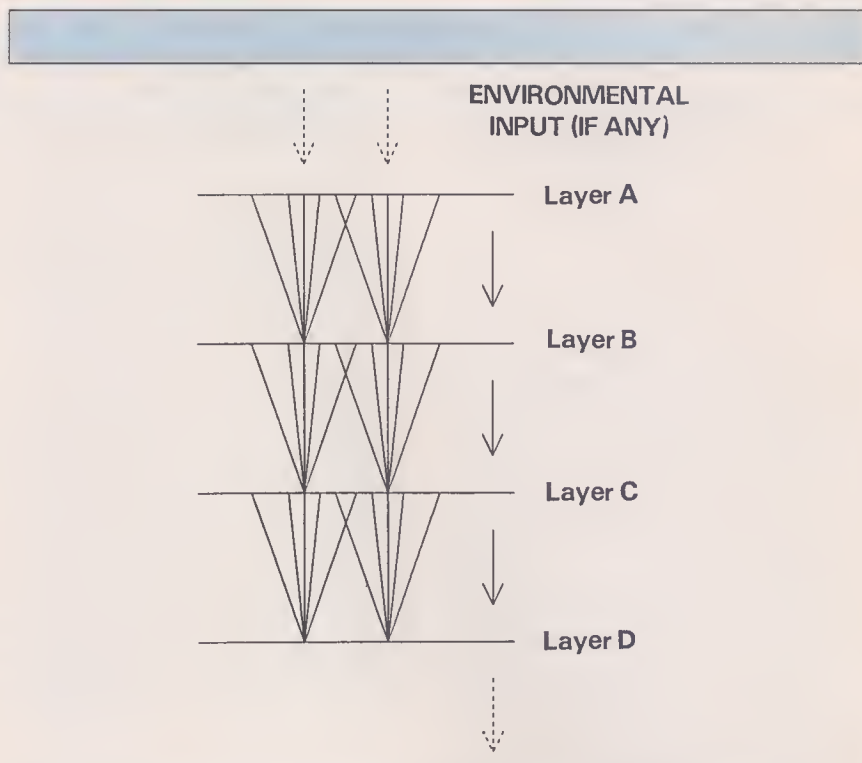


Figure 1. A layered self-adaptive network with local feedforward connections. Each two-dimensional layer contains many cells. Five input connections to each of two cells in layers B, C, and D are shown. Several hundred inputs to each cell are used in simulations. Each cell also provides input to many cells of the following layer. Lateral connections within a layer, as discussed in the text, are not indicated here.

previous work.¹

Consider a cell M and the cells $L_1, L_2, \dots, L_N$ that provide input to M . For simplicity, we avoid treating effects that depend upon the time sequence of signal activity values. Instead, we think of the activity history of a layer as a set of "snapshots," in which the ordering of the snapshots plays no role. That is, a set of activity values, denoted by $(L_1^n, L_2^n, \dots, L_N^n)$, is presented as input to the M cell, the M cell generates an output activity value M^n , and a new set of input activities is then presented. The superscript n indexes the presentation of inputs, that is, the particular snapshot, and the corresponding output. Then the linear response rule is

$$M^n = a_1 + \sum_j L_j^n c_j \quad (1)$$

where c_j is the strength of the j th input connection to the M cell. Our Hebb-type rule is

$$(\Delta c_j)^n = a_2 L_j^n M^n + a_3 L_j^n + a_4 M^n + a_5 \quad (2)$$

where the a 's are arbitrary constants ($a_2 > 0$). We assume that the c values change slowly from one presentation to the next. Then we can average Equation 2 over an ensemble of many presentations, and use Equation 1 to express M^n in terms of the $\{L_j^n\}$ to obtain the rate of change of each c value. Some algebraic manipulation¹ gives

$$\dot{c}_i = \sum_j Q_{ij} c_j + [k_1 + (k_2/N) \sum_j c_j] \quad (3)$$

where $k_{1,2}$ are particular combinations of the constants $a_{1,5}$. Apart from the determined values of $k_{1,2}$, the constants $a_{1,5}$ play no further role in what follows. Here

$$Q_{ij} \equiv \langle (L_i^n - \bar{L}) \times (L_j^n - \bar{L}) \rangle \quad (4)$$

is the covariance of the activities of input cells i and j , where $\langle \dots \rangle$ and the overbar both denote the ensemble average. (For our purposes, $\bar{L}$, the ensemble average of the input activity at a synapse, can

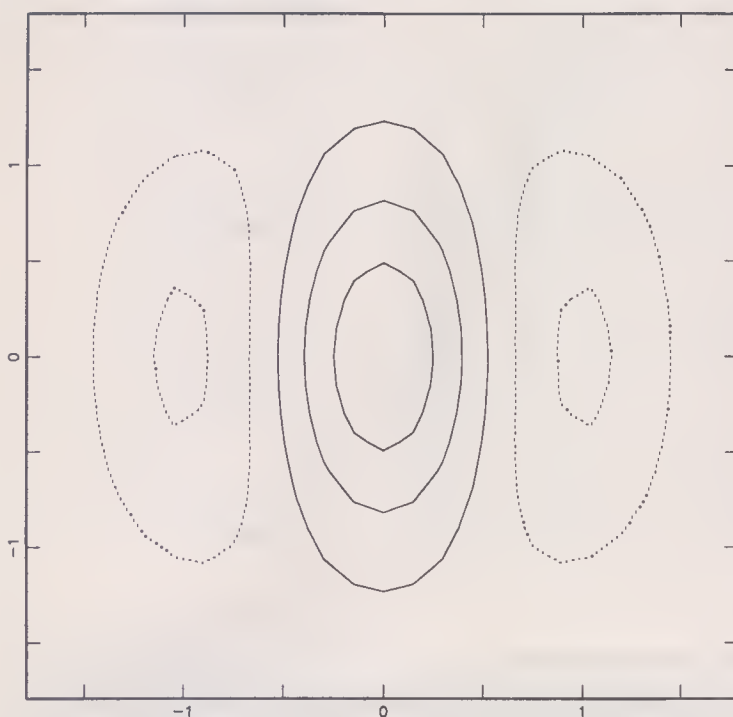


Figure 2. Receptive field map of a computed orientation-selective cell. A point of illumination at any position in the plane evokes an output response from the model cell that is proportional to the contour value at that position. Positive contour values (solid curves) denote an excitatory output response; negative values (dotted curves) denote an inhibitory response. Contour values range from -0.45 to $+0.75$ in steps of 0.30 . The peak response (at the receptive field center) is normalized to unity. The parameter values that generated this particular orientation-selective cell, and the units (r_G) of distance along the axes, are given in reference 1. (See Figure 1a, p. 8780). Axes denote distance of illumination point from receptive field center.

be taken to be the same for all synapses i, j .) The appearance of the input covariance matrix Q does not mean that there is any direct interaction between synapses i and j . Q appears simply because the Hebb rule causes $\dot{c}_i$ to depend upon the product $\langle L_i^n M^n \rangle$, and M^n in turn depends upon all the $\{L_j^n\}$ values (via Equation 1). The Q matrix will play an important role in what follows.

To prevent c values from becoming infinite during the development process, a saturation constraint is imposed. Each c value is constrained to lie between two values c_- and c_+ . In a more biologically realistic case, there are excitatory synapses that have $0 \leq c \leq c_+$ and inhibitory synapses that have $c_- \leq c \leq 0$. The analysis of this case gives the same result.

First the connections from layer A to B mature, or develop to their final values. That is, the initial c values are chosen at random, the set of differential equations given by Equation 3 (for $i = 1, 2, \dots, N$) is solved, using the Q_{ij} function that applies to layer A activity. (For random snow activity in layer A, Q_{ij} is 1 when i and j are the same A cell, and 0 otherwise.) Knowing the mature c values for the A-to-B connections, as well as the Q_{ij} function for layer A, then allows us to compute the Q_{ij} function for the mature layer B. Then the development of the B-to-C connections is computed, using the Q_{ij} function appropriate to layer B. By repeating the process, we compute in turn the connection strengths for successive layers of connections.

Simulation results. A few parameters for each layer of cells determine the mature c values of the cells in that layer. These parameters include k_1 and k_2 and the breadth of the region in the previous layer that provides input to a cell of the developing layer. (See Figure 1.) As we shall see, the choice of the $k_{1,2}$ values determines the mature value of the total connection strength Σc_j of the inputs to the M cell.

When we explore the parameter space, we find that there are a limited number of ways each layer can develop. Briefly, we find that a sequence of feature-analyzing cell types emerges as one layer after another matures.

The first cell type emerges in layer B. There is a parameter regime in which each c value reaches its excitatory limit c_+ . In this case, each B cell, once it has matured, computes the local average of the activity in the overlying region of layer A from which it receives input.

Once the B cells have matured in this way, nearby B cells have correlated activity. Each activity pattern in layer B is a blurred image of random snow. If one B cell's activity happens to be "high" at a given time, its neighbors' activities are likely to be "high" also. As a result of this activity correlation, a new cell type emerges in layer C. This *center-surround* cell type¹ acts as a contrast-sensitive filter—it responds maximally to a bright circular spot centered on the cell's receptive field, against a dark background. Center-surround cells having the reverse property—they respond maximally to a dark spot on a bright background—also emerge.

The Q function for pairs of center-surround cells in layer C determines the developmental possibilities for the C-to-D connections, and so on. We find that the next new type of feature-analyzing cell to emerge as we pass to succeeding layers is an *orientation-selective* cell. This cell responds maximally to a bright edge or bar against a dark background, or the reverse, when the edge or bar has a particular orientation. The receptive field map for such a computed cell is shown in Figure 2. This map is a contour plot showing the response of the cell to point illumination, as a function of the position of the illumination in visual space.

Each orientation-selective cell will develop to favor an arbitrary orientation if the network contains only feedforward connections as in Figure 1. However, if lateral connections between nearby cells of the orientation-selective cell layer are

included in the simulation, then the orientation preferences of the cells in the layer can become organized in certain arrangements. Cells having similar orientation preferences develop to occupy irregular band-shaped regions. (See reference 1 and the front cover, right side, of this issue.)

Discussion of the simulations. Center-surround cells are a prominent feature of mammalian retina. Orientation-selective cells emerge in cat and monkey visual cortex.^{3,4} Irregular band-shaped regions of cells of similar orientation—called *orientation columns*—are a prominent feature in the orientation-selective cell layers.^{3,4} (Once again, see the front cover, left side.) The role that lateral connections in cortex play in the formation of orientation selectivity is at present experimentally unsettled. As we noted, certain primates exhibit well-formed orientation selectivity at birth, in the absence of any structured visual experience.

Our point is not to suggest that feature-analyzing cells—particularly the center-surround cells—arise in animals in the same way they do in this synthetic network. As noted previously, the anatomy of inter-layer connections in the retina is more complex than a simple feedforward arrangement. Furthermore, center-surround cells can be constructed by a simple non-adaptive model in which excitatory inputs from some narrow region, and inhibitory inputs from a broader region, both converge on a cell. In our simulations we assumed that the breadth of the input region to a cell was the same for excitatory and inhibitory synapses, in order to avoid biasing the solution toward the formation of a center-surround cell type.

Our point is rather that a set of progressively more complex feature-analyzing cell types develops in the layered network, and that these cell types, and their organization, qualitatively exhibit some of the most salient features found in the first few stages of mammalian visual processing. The results suggest that some properties whose origin has been mysterious—such as orientation selectivity—may have a natural explanation in terms of the functioning of a Hebb-type development process in a layered network.

Two simple examples of how *structured* input to layer A would affect the simulation results are worth noting:

(1) If nearby pixels have correlated intensity values, and this is the only important input correlation present, then Q in layer A would resemble the Gaussian Q

that we found in layer B. The subsequent development of the model would proceed in a way similar to that which we described, except that the appearance of each feature-analyzing cell type could be advanced one layer.

(2) If layer A is shown an ensemble of patterns, each consisting of sinusoidal stripes with arbitrary phase and orientation, then orientation selectivity can develop as early as layer B.¹

We have assumed, for simplicity, that the statistical properties of the ensemble of presentations, that is, the covariances Q_{ij} , are unchanged or stationary during development. If the ensemble statistics change, cells that had reached their apparently final mature c values may change these c values in accordance with the new ensemble characteristics. Thus, although we always speak of cell *development*, the present approach is equally applicable to studying questions of cell *plasticity* during the life of the animal.

Hebb rules and optimization properties

We have seen that even a simple layered network with local feedforward connections obeying a Hebb-type rule develops a sequence of progressively more sophisticated feature-analyzing properties as we pass from one layer to the next. We will now examine some remarkable optimization properties of a Hebb-type rule.

Maximization of output activity variance. Consider a cell M that receives input from cells $L_1, L_2, \dots, L_N$. Here and later, “input” means local input to cell M , not the environmental input to the network as a whole. Similarly, “output” refers to the M cell’s activity value, not to the output from the network as a whole. Let the M cell’s development be described as in Equations 1-4, with a saturation constraint on the range of each c value. We assume that the ensemble statistical properties of the L -cell activities, that is, the Q_{ij} function for the L cells as in Equation 4, are unaffected by the choice of c values. This is true if there is no feedback from M , or the cells it influences, to the L cells. It should be a satisfactory approximation if the feedback is present but is sufficiently weak, although this has not been studied quantitatively.

Define the function

$$E \equiv E_Q + E_k \quad (5)$$

where

$$E_Q \equiv -(1/2) \langle (M^n - \bar{M})^2 \rangle \\ = -(1/2) \sum_i \sum_j Q_{ij} c_i c_j \quad (6)$$

and

$$E_k \equiv -k_1 \sum_j c_j - (k_2/2N) (\sum_j c_j)^2 \quad (7)$$

I have constructed the function E to have the property that $-\partial E / \partial c_i = \dot{c}_i$ for each i . This means that, as the Hebb rule causes each of the c values to change with time, the value of E , as a function of the c ’s, decreases along a path of locally steepest, or gradient, descent. (If $\dot{c}_i > 0$, then $\partial E / \partial c_i < 0$, so c_i increases and E decreases with time. If $\dot{c}_i < 0$, then $\partial E / \partial c_i > 0$, so c_i decreases and E again decreases with time.)

The value of E thus achieves a local minimum at cell maturity. Moreover, for the cases of interest here—including those that lead to the center-surround and orientation-selective cell types—this minimum is a *global* near-minimum as well.¹ We therefore will focus on the case in which the development process does not get stuck in high-lying local minima. This appears to be the typical case for a perceptual network exposed to a large ensemble of presentations, although it is an empirical finding and I have not established the limits of its validity.

What is the meaning of E achieving a global, or absolute, minimum value? For any given value of total connection strength $\sum c_j$, E is minimized when $\langle (M^n - \bar{M})^2 \rangle$ —the statistical variance of M —is maximized. Changing the values of the parameters $k_{1,2}$ adjusts, or tunes, the mature value of $\sum c_j$. The E_k term, which is a function of $\sum c_j$ and $k_{1,2}$ only, plays a role similar to a Lagrange multiplier term, although E_k is parabolic rather than linear in $\sum c_j$.

Therefore, the development rule of Equation 3 causes a cell to develop so as to maximize the variance of its output activity, subject to the constraint that the total connection strength have a given, parameter-determined, value and subject to the saturation bounds for each c value. Let us see intuitively what variance maximization means to a perceptual system.

Consider first a hypothetical M cell whose c values are such that the cell’s output variance is zero. That is, regardless of the input values ($L_1^n, L_2^n, \dots, L_N^n$) chosen from the ensemble of presentations,

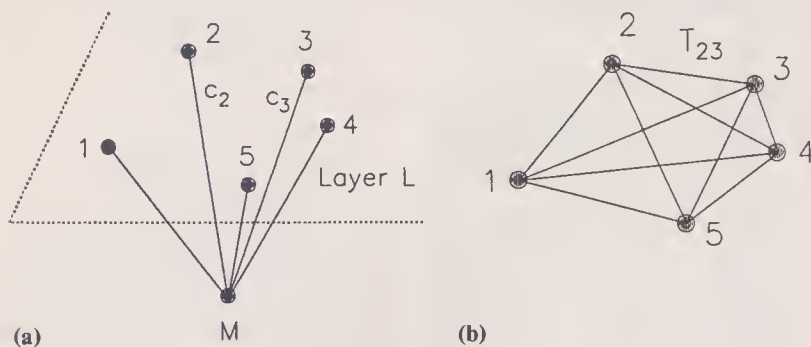


Figure 3. Relationship between networks: (a) a single M cell (with N inputs) of a layered self-adaptive network; (b) a Hopfield network with N cells and $N(N-1)/2$ connections, where $N=5$.

the output is always the same. This cell would be useless for conveying any information about the environment to later parts of the perceptual system.

On the other hand, if the c values are chosen in a different and special way, then the M cell's output value exhibits the largest possible spread or variance, consistent with the constraints on the c 's, as the set of input values ranges over its ensemble. We have shown that a Hebb-type rule tends to generate c values satisfying this special condition. In an informal sense, provided certain conditions are met, the Hebb rule acting on our described M cell tends to produce an M cell whose output activity optimally preserves the information contained in the set of input activities. Later, we will make this statement more precise, by applying some concepts from information theory, and we will modify it to accommodate the situation in which multiple M cells interact with one another.

Optimization in another type of neural network. Hopfield⁵ emphasized that the dynamics of a neural network can be described in some cases by the local minimization of a function. An interesting mathematical relationship exists between the E function defined in Equations 5-7 and Hopfield's energy function—although the network structure and behavior that each describes are very different.

Once again, our E function is $E \equiv E_Q + E_k$ where E_Q is shown in Equation 6. The development rule causes E_Q to be minimized subject to the constraint that Σc_j have a specified value and subject to the saturation constraints on each c value. The arrangement described by Equation 6 consists of one M cell with N inputs from cells $L_1, L_2, \dots, L_N$ and is shown in Figure 3a. The $N \times N$ matrix of elements Q_{ij} is the covariance matrix of the input cell activities. The c 's are the connection strengths from each input cell to the output cell. The minimization of E describes the development of the c 's under the influence of the ensemble of inputs characterized by the covariance matrix Q .

In Hopfield's case,⁵ as illustrated in Figure 3b, there are N cells and the activity state of the i th cell is called V_i . Each pair of cells is connected with fixed connection strength T_{ij} , so the number of connections is of order $N^2/2$, and the energy function is

$$E' \equiv -(1/2) \Sigma_i \Sigma_j T_{ij} V_i V_j \quad (8)$$

The activities V_i change with time according to a linear summation rule with a threshold: V_i increases, unless it is already at its upper limit, if $\Sigma T_{ij} V_j > 0$, and decreases if $\Sigma T_{ij} V_j < 0$. If the T_{ij} matrix is symmetric, then the V_i 's change so as to decrease the value of E' to a local minimum. Connection strengths are fixed;

there is no learning or network development. The dynamical process described by Equation 8 is the change in the activities $\{V_i\}$ from some initial state to a final state of locally minimum E' . If we want to use the network for memory retrieval, a suitable choice of T_{ij} is given by an expression that is essentially the covariance of V_i^k and V_j^k over the ensemble of memories, indexed by k , to be stored.

Note that E' has the identical structure as our E_Q , if we identify V_i with c_i and T_{ij} with Q_{ij} . When T is a covariance matrix, Hopfield's network computes a local minimum of E' using N cells and order $N^2/2$ connections, explicitly embodying the T values. The state for which E' is minimal is the set of final activities $(V_1, V_2, \dots, V_N)$.

One cell of our network computes a local minimum of the same function, our E_Q , using N connections. The Q function, which corresponds to T , is nowhere explicitly represented in the network. The Hebb rule implicitly responds to the covariance matrix, Q , as the ensemble of input patterns is presented to the M cell. The state for which E_Q is minimal is not a set of activities, but a set of mature connection strengths $(c_1, c_2, \dots, c_N)$.

Thus, for T matrices that are covariance matrices, one cell of our network can locally optimize the same function as a fully connected Hopfield-type network. In our network, this optimization process consists of developing a final set of c values, starting with some initial set of values, under the influence of a statistically stationary ensemble of input patterns having covariance matrix T . In the Hopfield-type network case, the process consists of seeking a final set of cell activity values starting with some initial set of values, in a network whose connection strengths are fixed and prespecified to be the T values themselves.

These considerations lead to an interesting connection, only briefly outlined here, between memory retrieval and perception in a network model.

Memory retrieval and perception in a network model. If there are sufficiently few memory patterns to be stored, relative to N , then E' or E_Q will tend to have minima at the $\{V_i\}$ or $\{c_i\}$ values, respectively, corresponding to those memories. Depending upon the initial choice of the V 's or c 's, one or another of these memory states will be activated or selected. In the case of Hopfield's network, "activated" means that the final activity state

will match one of the stored memories. In the case of a cell in a layered self-adaptive network, "selected" means that the final set of c values will cause the M cell to be a matched filter for one of these memories. That is, the mature M cell will respond most strongly when presented with the set of input activities corresponding to that memory.

If the number of patterns in the ensemble is large, then the E_Q function will no longer capture details of any one of the patterns. The structure of the E_Q function may become simpler. The global minimum of E_Q will lie at the $(c_1, c_2, \dots)$ value for which the M cell's variance is maximized. The mature M cell will function as a feature-analyzing cell, rather than as a matched filter to a particular memory. The particular feature or pattern element to which the mature cell will optimally respond, such as an oriented edge, need not even appear in any of the presented patterns.

Principal component analysis. There is a special case in which variance maximization corresponds to an important, and widely-used, statistical method for feature extraction. This is the case in which the output variance is maximized subject to the constraint that $\sum c_i^2 = 1$. Oja⁶ showed that this maximization can be achieved by using a particular form of the Hebb rule, equivalent to

$$\dot{c}_i \propto \langle M^n (L_i^n - M^n c_i) \rangle \quad (9)$$

For this expression, we put $M^n \equiv \sum L_i^n c_i$ and define the activities, subtracting non-zero mean values if necessary, so that $\langle L_i^n \rangle = 0$ for all i . The additional term in the Hebb-type rule, proportional to c_i , causes $\sum c_i^2$ to be close to 1, and no explicit constraint needs to be imposed.

In statistics, principal component analysis, or PCA, is a standard method, reviewed in Huber,⁷ for identifying "interesting" but unanticipated structure, such as clustering, in high-dimensional data sets. For example, an economist confronted with 1000 dimensions of data, such as the prices of different commodities, may want to know which several features of the data, for example, which several linear combinations of the 1000 quantities, are most salient.

PCA works as follows. Consider a set of data points indexed by π , each point L^π having coordinates $(L_1^\pi, L_2^\pi, \dots, L_N^\pi)$. For PCA, we compute a vector c for which the projection of the set of data points

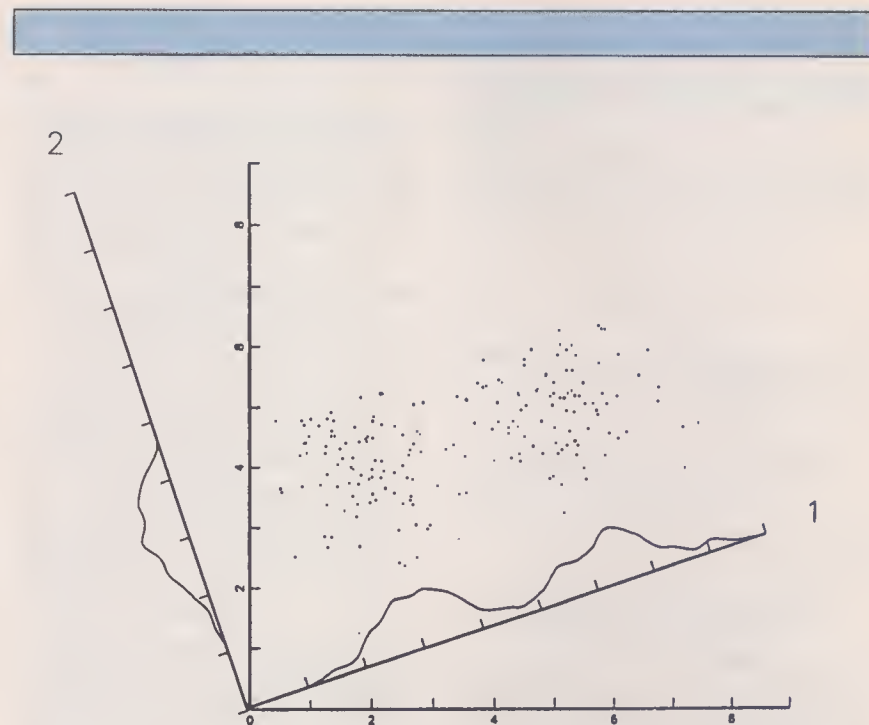


Figure 4. Illustration of principal component analysis. A cloud of data points is shown in two dimensions, and the density plots formed by projecting this cloud onto each of two axes 1 and 2 are indicated. The projection onto axis 1 has maximum variance, and clearly shows the bimodal, or clustered, character of the data.

onto the axis parallel to c has maximum variance. The projection of L^π onto c , when $\sum c_i^2 = 1$, is just $M^\pi = \sum L_i^\pi c_i$, and the variance of the projected distribution is identical to the variance of M^π .

An example of PCA is illustrated in Figure 4. Projecting the cloud of data points onto line 1 captures the salient feature of the data—that there are two clusters. The variance, or spread, of the data points along this axis is greater than for any other projection axis. Projecting the cloud onto line 2 would obscure the cluster structure. While the cluster structure is evident in the raw data of the two-dimensional plot shown here, such structure is often totally concealed in high-dimensional data sets, until an analysis method such as PCA is applied.

Since the PCA method corresponds to choosing c so as to maximize the variance of M^π subject to $\sum c_i^2 = 1$, it follows that the mature M cell generated by Oja's version of the Hebb rule performs PCA on its set of inputs.⁶

Optimal inference. Consider an arbitrary M cell characterized by a set of c values and having the linear response rule $M^\pi = \sum L_i^\pi c_i$ with $\langle L_i^\pi \rangle = 0$ for all i . Suppose we know the c values, and are told a particular value of the output, M^π . We are asked to estimate the input activities $(L_1^\pi, L_2^\pi, \dots, L_N^\pi)$ for that presentation. Let us score any such estimate by

- (1) computing the difference between the estimate $L_i^\pi(\text{est})$ and the true value of L_i^π
- (2) squaring this difference, and
- (3) summing this squared error over i .

Averaging this score over an ensemble of presentations gives the mean square error

$$\text{MSE} \equiv \sum_i \langle [L_i^\pi - L_i^\pi(\text{est})]^2 \rangle \quad (10)$$

What estimation rule will give the best, meaning the minimum, MSE? For a linear estimation rule of the form $L_i^\pi(\text{est}) \equiv g_i M^\pi$, where we want to know what g values to use, the answer is found by

minimizing MSE with respect to each of the g_i 's. This is easily done by differentiating MSE. It is also a simple case of the Gauss-Markoff theorem,⁸ which applies more generally to the optimal estimation of a set of inputs given a set of outputs, rather than just one output. The result is

$$L_i^n(\text{opt est}) = M^n \times (\sum_j Q_{ij} c_j) / (\sum_i \sum_j c_i Q_{ij} c_j) \quad (11)$$

The MSE corresponding to this optimal estimate is then

$$\text{MSE}(\text{opt}) \equiv \sum_i < [L_i^n - L_i^n(\text{opt est})]^2 > = \sum_i < (L_i^n)^2 > - H \quad (12)$$

where

$$H \equiv [\sum_i (\sum_j Q_{ij} c_j)^2] / (\sum_i \sum_j c_i Q_{ij} c_j) \quad (13)$$

Expressed in matrix form, with $\mathbf{c}$ denoting the column vector $(c_1, c_2, \dots)$ and $\mathbf{Q}$ denoting the matrix (Q_{ij}) , we have $H \equiv (\mathbf{c}^T \mathbf{Q} \mathbf{Q} \mathbf{c}) / (\mathbf{c}^T \mathbf{Q} \mathbf{c})$, where the superscript $\mathbf{T}$ denotes the matrix transpose.

The calculation so far involves a standard use of optimal estimation theory.⁸ The linear filter, represented here by the set of c values, is specified. The result of a measurement using the filter—that is, the output value—is given. The task is to reconstruct the input values with minimum error, using a simple mean squared error criterion.

We now go beyond this simple framework to ask:⁹ For what linear filter—what set of c values—is this minimum-error reconstruction the most accurate? That is, what choice of c 's minimizes $\text{MSE}(\text{opt})$ of Equation 12?

Since the first term on the right-hand side of Equation 12 is independent of the c 's, minimizing $\text{MSE}(\text{opt})$ is accomplished by maximizing H . The mathematical condition for this to occur is that the vector $\mathbf{c}$ be an eigenvector of $\mathbf{Q}$ having maximal eigenvalue. This is identical to the condition that $\mathbf{c}$ needs to satisfy in order for the M cell to perform PCA on its input values.

Therefore the PCA condition and the principle of optimal inference—namely, that $\text{MSE}(\text{opt})$ be minimized, or H be maximized—lead to the same set of c values. A Hebb rule of the form of Equation 9 generates an M cell that satisfies both conditions. In the presence of other constraints, or additional cost terms, there is no guarantee that PCA and H-maximization are equivalent, since the PCA principle maximizes the quantity

$(\mathbf{c}^T \mathbf{Q} \mathbf{c}) / (\mathbf{c}^T \mathbf{c})$ which is not identical to the expression for H in Equation 13.

Optimization in the presence of processing noise and constraints on output variance. We have identified several optimization properties related to the cell's output variance. Suppose, however, that for some reason the variance is itself constrained. For example, the output activity may be confined to lie within some operating range. This is a biologically plausible situation. In this case, what is optimized by a suitable Hebb-type rule? We will discuss this case for a particular processing model, giving only the main results and omitting the details.

Suppose the signal L_j^n on the j th input line or connection is corrupted by noise, v_j^n , where v_j^n has a mean of zero and a variance B , and is uncorrelated both with the noise on other input lines and with any of the input signals L_j^n . The cell computes the weighted sum $x \equiv \sum_j (L_j^n + v_j^n) c_j$. The variance of x is the sum of two terms: the variance due to the signal in the absence of noise, $\sum_{ij} Q_{ij} c_i c_j$; and the variance due to the noise, $B \sum_j c_j^2$. Consider a suitable synaptic modification rule in which $\dot{c}_i$ contains a term of the form $< L_i x >$. This rule causes the model cell to develop such that the variance of x due to the signal is maximized relative to the variance due to the noise. This type of signal-to-noise optimization property can also emerge when the cell's output M is a monotonic nonlinear function of x , such as a sigmoid function, if the synaptic modification rule is of the form described.

Adaptive signal processing. Returning to the case of a linear-response model neuron, suppose we wish to train a linear cell to respond to each of a set of prescribed input vectors by generating an output that best matches a prescribed desired output. An input vector is denoted $\mathbf{L}^n \equiv (L_1^n, L_2^n, \dots, L_N^n)$ and each desired output is a scalar number M_{des}^n . The actual output is $M^n = \sum_j L_j^n c_j$ where each $< L_i^n > = 0$ and the optimal c values are to be determined by a learning process. A mean square measure of error is used:

$$\text{MSE}' = < (M^n - M_{\text{des}}^n)^2 > \quad (14)$$

where $< \dots >$ again indicates the ensemble average. MSE' is a minimum when the c values are chosen to satisfy $< M_{\text{des}}^n L_i^n > = \sum_j Q_{ij} c_j$ for all i . (Recall that $Q_{ij} \equiv < L_i^n L_j^n >$.) The least mean square, or LMS, algorithm of Widrow and

Hoff¹⁰ uses an estimate of the gradient of MSE' and in effect performs gradient descent to compute the optimal c values. An ensemble-averaged form of the algorithm can be written as

$$\dot{c}_i \propto < L_i (M_{\text{des}}^n - \sum_j L_j^n c_j) > \quad (15)$$

Equations 14 and 15 give an objective function to be minimized and an algorithm for a supervised learning process. Both the inputs and the desired outputs are presented to the cell, and the error term $(M_{\text{des}}^n - M^n)$ —the amount by which the actual output differs from the desired output—is fed back to change the c values until the mean square error is minimized.

Our optimal inference criterion, namely, the minimization of the objective function of Equation 12, and a Hebb-type rule that implements it (Equation 9) are formally similar to Equations 14 and 15. But the optimal inference criterion provides a method for unsupervised learning. The criterion does not make any use of a desired output; it simply states that the M cell should have the property that knowing its output activity value allows one to infer the input activities with greatest possible accuracy.

Information theory and the principle of maximum information preservation

For a single M cell receiving inputs from a given set of L cells, we have seen that, for a particular Hebb rule given in Equation 9, knowledge of the output activity value allows inference of the input values with greatest accuracy, in the sense of minimum mean squared error. For more general Hebb-type rules, we found that the variance of the output activity was maximized subject to various constraints. This result led us to suggest that, at least in an intuitive sense, a Hebb rule may act to generate an M cell whose output activity preserves maximum information about the input activities, subject to constraints.

We will now make this notion of maximum information preservation more precise, and will extend it to the case of an entire layer of M cells, by introducing some concepts from information theory. The goal is to see what this principle implies for the development of each layer of a perceptual system. That is, given the statistical properties of the ensemble of

input patterns at layer L , and certain constraints, what particular processing functions do the connections from layer L to layer M , and within layer M , develop to implement?

Shannon information. We will regard each presentation of real-valued inputs $L = (L_1, L_2, \dots, L_N)$ as a message, where L_i denotes the activity of the i th L cell in the layer. We omit the π superscripts for clarity. Strictly speaking, even one real number carries an infinite amount of information. To avoid encountering expressions of the form, $\infty - \infty$, and because infinite precision is physically and biologically meaningless, we will think of the N -dimensional space of the L vectors as being divided into small boxes. Each box is labeled by its location L . Two messages are regarded as identical if they lie in the same box. In the end, we will pass to the continuum limit, and the sums will become integrals.

Given an ensemble of messages, let $P(L)$ be the probability that a randomly chosen message lies in box L . Shannon¹¹ showed, in a classic paper, that the information conveyed by sending a message that lies in box L is $I(L) = [-\ln P(L)]$. The average information conveyed per message is $\langle [-\ln P(L)] \rangle = -\sum_L P(L) \ln P(L)$, where $\langle \dots \rangle$ is the usual ensemble average. If the base-2, rather than natural, logarithm were used here, the information would be measured in bits.

Now suppose each input presentation L generates a set of output values, denoted by the vector M , via some known computation. Suppose that we are told the value of M —or, more strictly, which discrete box M lies in. (In general, M will not be uniquely determined by L because noise may be introduced in the computation of M .) How much additional information would we need to reconstruct the input message L that gave rise to M ? (Shannon calls the ensemble average of this amount of additional information the equivocation.)

The answer is $I_M(L) = [-\ln P(L|M)]$, where $P(L|M)$ is the conditional probability of the input message lying in box L given that the output lies in box M . Therefore the amount of information that knowing M conveys about L is the difference, $I(L) - I_M(L) = \ln[P(L|M)/P(L)]$. The ensemble average of this quantity is the rate R , per message, of transmission of information from the cell's inputs to its output. This is the average amount of information that knowing M conveys

about L . We have

$$R = \langle \ln[P(L|M)/P(L)] \rangle \quad (16)$$

We have a standard identity $P(L|M)P(M) = P(L, M) = P(M|L)P(L)$, where $P(L, M)$ is the joint probability that the input lies in box L and the output lies in box M . Using this gives

$$\begin{aligned} R &= \langle \ln[P(M|L)/P(M)] \rangle \\ &= -\langle \ln P(M) \rangle + \langle \ln P(M|L) \rangle \\ &\equiv I(M) - \langle I_L(M) \rangle \end{aligned} \quad (17)$$

The right-hand side is the ensemble average of the total information conveyed by M , minus the information that M conveys to one who already knows L . This second term is the "information" that M conveys about the processing noise, rather than about the signal L .

Maximum information preservation. Let us now state the proposed principle of maximum information preservation for each layer, or processing stage, of a perceptual network: Given a layer L of cells, and the stationary ensemble statistical properties of the signal activity values in the layer, and given that layer L is to provide input to another cell layer M , the transformation of activity values from L to M is to be chosen such that the rate R of information transmission from L to M is maximized, subject to constraints and/or additional cost terms. These constraints or costs may reflect, for example, biochemical and anatomical limitations on the formation of connections, or on the character of the allowed transformations.

The formulation of this principle arose from studying Hebb-type rules and recognizing certain optimization properties to which they lead for single M cells. Once formulated, however, the principle is independent of any particular local algorithm, whether Hebb-related or otherwise, that may be found to implement it. Let us explore

- (1) the consequences of the principle for some simple cases;
- (2) how the principle might be implemented; and
- (3) how it may fit within a broader view of neural development.

A single M cell. Under certain conditions, maximizing the output activity variance of the M cell maximizes the Shannon information rate R . We illustrate this for a particularly simple but instructive case. The argument can be made somewhat

more general than this, but it is not true that maximum information rate and maximum activity variance coincide when the probability distribution of signal values is arbitrary.

Suppose the M cell receives inputs from a set of L cells $L_1, L_2, \dots, L_N$, and that the M cell's output in the presence of processing noise has the form

$$M^\pi = (\sum_i L_i^\pi c_i) + v^\pi \quad (18)$$

Here π indexes the particular set of input and output values, so that if L is repeated but the output M is different, owing to noise, this counts as a different set of input-output values. The quantity v^π is the noise, a random variable differing from one presentation to the next. Suppose that

- (1) M has a Gaussian distribution, with variance denoted by V ;
- (2) v has a Gaussian distribution with a mean of zero and variance denoted by B ; and
- (3) v is uncorrelated with any of the input components; that is, $\langle v L_i \rangle = 0$ for all i .

Then, omitting the details, we find that the information rate is

$$R = (1/2) \ln(V/B) \quad (19)$$

For a given noise variance, B , this rate is maximized by maximizing the output variance V of the M cell. Note that V/B is essentially a signal-to-noise ratio.

Suppose that the noise model consists instead of independent Gaussian noise, v_i , being introduced on each input line i , where each v_i has variance B . Then $M^\pi = \sum_i (L_i^\pi + v_i^\pi) c_i$, and the information rate is found to be $R = (1/2) \ln[V/(B \sum c_i^2)]$. In this case, R is maximized for fixed B when $(V/\sum c_i^2)$ is maximized—that is, when the connection strengths are chosen so as to perform principal component analysis on the cell's inputs.

Redundancy and diversity. Suppose there is an arbitrary number of L cells but just two coupled linear M cells. Each M cell's output is some linear combination of the L cell's activities:

$$M_1^\pi = (\sum_i t_{1i} L_i^\pi) + v_1^\pi \quad (20)$$

$$M_2^\pi = (\sum_i t_{2i} L_i^\pi) + v_2^\pi \quad (21)$$

Each noise term is Gaussian and of variance B , the noise terms for the two M cells are uncorrelated with each other, and each noise term is uncorrelated with any of the

L cell activities. We treat the case in which M_1 and M_2 have Gaussian distributions, with $\langle M_1^n \rangle = \langle M_2^n \rangle = 0$. Our task is to determine what values of the t_{ni} 's lead to the maximum information being preserved during the processing of L-cell activities to give M-cell output activities.

Note that the t_{ni} 's do not in general stand for the strengths of particular connections. There may be both feedforward and lateral (M-to-M) connections whose joint effect, possibly over several time steps, is to produce the M-cell outputs of Equations 20 and 21. Our concern here is not with the particular connection strengths, nor with the development rule that may implement them, such as a Hebb-type rule, but rather with understanding what cell response properties—what t_{ni} values—are induced by the principle of maximum information preservation.

Omitting details of the proof, the resulting information rate for this case is

$$R = (1/2) \ln(\text{Det } Q^M) - \ln B \quad (22)$$

where the elements of the 2×2 covariance matrix Q^M are $Q_{nm}^M = \langle M_n^M M_m^M \rangle$ and "Det" denotes the determinant. We find

$$\text{Det } Q^M = B^2 + B(W_1 + W_2) + W_1 W_2 (1 - \rho_{12}^2) \quad (23)$$

where W_n is the output variance of cell M_n in the absence of noise, and ρ_{12} is the correlation coefficient of the activities of M cells 1 and 2, also in the absence of noise.

To maximize R , given B , we must maximize $\text{Det } Q^M$. When B is large, the third term on the right-hand side of Equation 23, which is independent of B , is small compared with the second term, which is of order B . In that case, maximizing $\text{Det } Q^M$ means maximizing $(W_1 + W_2)$. If no constraint prevents us, we can achieve this maximization by maximizing W_1 and W_2 separately. But this means constructing each M cell so that its output variance, which is W_n in the absence of noise, or $W_n + B$ in the presence of noise, is maximized. This is exactly what we found to be the optimum solution when there is only one M cell. (See Equation 19.)

If the noise B is smaller, then the third term becomes relatively more important. The rate R is then maximized by making an optimal tradeoff between keeping W_1 and W_2 large, and making the responses of the two M cells uncorrelated.

We have thus found that, depending upon the noise level, there is competition between the value of having redundant M

cell responses, which mitigate the information-destroying effects of noise, and the informational value of having different cells extract different linear combinations of the input. A high noise level favors redundancy. In this case, both M cells compute the same linear combination of inputs, if there is only one such combination that yields maximum output activity variance. A lower noise level favors diversity of response. In this case, the M cells compute different linear combinations of the L cell activities, even though each M cell's output variance may be reduced as a result of this choice.

To make this more concrete, consider a simple example. There are two L cells, and the Q matrix for L cell activity has $Q_{11} = Q_{22} = 1$ and $Q_{12} = Q_{21} = q$ with $0 < q < 1$. We arbitrarily impose the constraint that $t_{n1}^2 + t_{n2}^2 = 1$ for each M cell ($n = 1, 2$).

The solution that maximizes the preservation of information then has $t_{11} = t_{22}$ and $t_{12} = t_{21}$, and the values of t_{11} and t_{12} are given in Figure 5 as a function of B and q . For large B , both M cells receive the same linear combination of inputs: $(L_1 + L_2)/\sqrt{2}$. For smaller B , the cells measure different linear combinations of L_1 and L_2 . In the limit as B approaches zero, one M cell receives input only from cell L_1 and the other only from L_2 .

A layer of M cells with nonlinearity and lateral connections. What does the principle of maximum information preservation, which we shall call the *infomax principle*, imply qualitatively in this more general case? Maximizing R means that we attempt to (1) maximize the total information conveyed by the output message **M**, and (2) minimize the information that **M** conveys to one who already knows the input message **L**. These criteria are related, but not equivalent, to the property of encoding signals so as to reduce redundancies present among the inputs to the perceptual system. The general idea that information theory can be useful for understanding perception is an old one. Significant contributions were made by Attneave in 1954, Barlow in the 1950s and 1960s, and Marr in 1970. Much of this work has focused on the role of redundancy reduction. This property is one, but only one, aspect of the infomax principle. For example, we have seen that infomax also leads to the introduction of redundancy when this is useful in countering the effects of noise.

I have analyzed the qualitative conse-

quences of the infomax principle in some very simple models.¹² The results show that the principle can, under certain conditions, lead to L-to-M transformations with the following properties:

- Topographic mapping from layer L to layer M, when the spatial extent of lateral connections within layer M is assumed to be limited. That is, near-neighbors in L tend to map to near-neighbors in M.
- Map distortions, in which a greater number of M cells tend to represent the types of layer-L excitation patterns that occur more often.
- The infomax principle selects which features of the input signals are represented in layer M. Features having relatively high signal-to-noise ratios are favored. This is the extension of our previous redundancy-diversity result to the full-layer case.
- Orientation-selective cells, and the arrangement of such cells in orientation columns, can emerge for some very simple types of model input.
- When time-delayed information is made available to the layer, the infomax principle can cause M cells to extract and encode temporal correlations, in a manner similar to the extraction of spatial correlations.

I must emphasize that much work is required to determine the consequences of the infomax principle for cases involving more biologically realistic patterns of activity.

Discussion

From a simplified set of assumptions—a linear summation response, a simple Hebb-type rule having a covariance form, and feedforward connections only—we derived an optimization principle for the development of a single cell. This principle states that the mature M cell is such that its output activity variance is maximized subject to constraints. More generally, we can have cost terms instead of, or in addition to, constraints. Then the function maximized involves both the variance and the additional cost function.

This led us to infer a proposed principle of maximum information preservation, subject to constraints. It is equivalent to variance maximization in some simple cases, but it has a much broader scope. For example, it can be applied to cases in which a layer of L cells provides input to an entire

layer of M cells, with lateral as well as feed-forward connections. It can likewise apply to cases in which the response function is not necessarily linear.

The consequences of this proposed principle are only beginning to be explored. One set of issues that needs clarification is the choice of biologically appropriate constraints and cost terms. A second, related, issue involves the choice of algorithms, whether of Hebb type or otherwise, that control the development of feedforward and lateral connections so as to implement the optimization principle. While much work needs to be done, I suggest that this principle, or something like it, may play an important role in determining the character of perceptual processing at least in its early stages, where there is a chance that feedback influences may not affect the development of feature-analyzing function in an essential way. Possibly the principle may play some role even in the presence of significant feedback, but it is not clear at this time how best to analyze this case.

What might we expect to be the character of a layer of cells developing according to the principle of maximum information preservation, for cases of biological interest? Although the necessary calculations for sufficiently realistic cases have not yet been carried out, we can speculate on the outcome.

Suppose there is a constraint on the distance within layer M over which the activity of one M cell can affect another. There might be, for example, a constraint on the length of lateral connections. Suppose also that each region of layer M "sees", or receives input from, only a limited region of layer L, and that nearby regions of M "see" nearby regions of L. Then, if the noise variance B is large, and there are not many M cells that "see" the same set of L cells, we might find that each M cell develops so as to maximize its activity variance, and performs processing that is redundant with that of many of its neighbors.

On the other hand, if B is smaller, or there are a large number of M cells that "see" the same L region, we may expect that the M cells in a region do not all perform the same processing function on the inputs from the L layer. Instead, they might span a range of feature-analyzing properties, each of which has a moderately high variance.

In the visual system of cats and monkeys, there are multiple layers of center-surround cells, followed by layers of

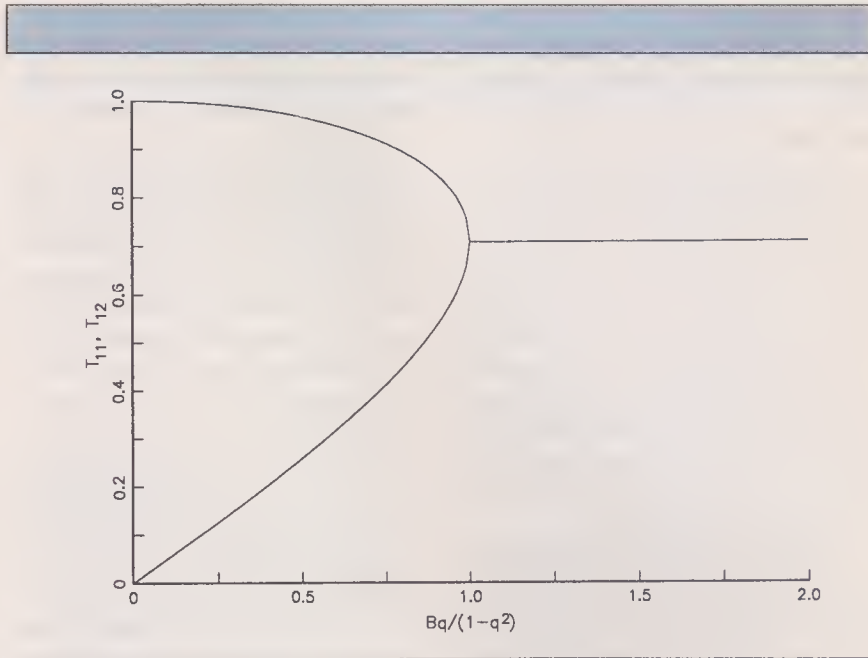


Figure 5. Values of the coefficients t_{ni} that maximize the preservation of information from layer L to M, for a simple case with two L and two M cells. Each M cell output (see Equations 20 and 21) includes random noise having variance B , and q is the correlation, or covariance, of the activities of the two L cells. For $x \equiv Bq/(1-q^2) \geq 1$, both M cells redundantly compute the same linear combination of the L cell activities (all $t_{ni} = 1/\sqrt{2}$). For $x < 1$, the optimal t values satisfy $t_{11} = t_{22}$ and $t_{12} = t_{21}$, where the upper curve gives t_{11} and the lower curve gives t_{12} , or the reverse. The curves for $x < 1$ are given by $y = (1/2)[(1+x)^{1/2} \pm (1-x)^{1/2}]$; this is derived by maximizing $\text{Det } Q^M$. (See Equation 23.)

orientation-selective cells. The orientation-selective cells begin at a different layer in cats than in monkeys. It is possible that, in response to the ensemble of inputs seen by a particular layer, the layer can develop either center-surround or orientation-selective cells, as occurred in our previous model simulations.¹ Perhaps a parameter such as the noise level B "tunes" for redundancy or diversity of response. Redundancy could favor center-surround cell formation, with many cells performing substantially the same processing function. Diversity could favor the formation of orientation-selective cells spanning the entire range of orientation preferences within each region of the layer. (Of a group of cells comprising all orientation preferences, only a small fraction will fire when presented with an oriented edge of illumination.) Hubel and Wiesel discovered^{3,4} that orientation-selective cells are arranged, within a cortical layer, so that each small region of cortex ($\approx 1 \times 1$ mil-

limeter) contains the "machinery" for analyzing substantially all edge orientations seen by either eye within a small region of visual space. Perhaps the principle of maximum information preservation, combined with limits on lateral interaction distance, can account for this efficient organization.

Local algorithms. The infomax principle is stated in terms of maximizing a complicated expression (see Equation 16). Is there an algorithm or process that deals with much simpler quantities and computations—local to each cell or pair of connected cells in a network—and yet implements the infomax principle, at least approximately?

I have found¹² that, for some simple cases, a Hebb-related algorithm developed by Kohonen¹³ implements some of the qualitative features required by the infomax principle. This algorithm was developed to show how lateral connections can

induce topographic order in a simple model, and makes no reference to noise or information content. These results suggest that it may be possible to devise a local algorithm that more fully embodies the requirements of the infomax principle.

The relationship between the principle and such an algorithm would be complementary. The principle would suggest what the function of the algorithm, and the lateral connections it describes, might be—that is, what role the processes and connections might serve in the construction of a perceptual system. The algorithm would show how a complex optimization principle could be implemented by a network of cells that individually have little computational power.

Although I have focused on algorithms that perform activity-dependent modification of connections, other types of mechanisms may be used to implement a given optimization principle. Biochemical cell-cell adhesion markers, chemical or other gradients that may help to establish topographic maps, particular cell types that implement complex types of connectivity (as in the retina), and other mechanisms may all play a role. An organizing principle by itself does not determine the many design details that a particular system—biological or synthetic—may use to implement it.

Infomax and perceptual data. Why might it be important for a perceptual system to maximize the amount of information preserved from one layer to the next?

Presumably, one goal of a perceptual system is to provide the brain with the means of discriminating different environmental situations that may demand different responses by the animal.

For a very simple network with only a couple of layers of processing from environmental input to motor output, we could imagine using some sort of supervised learning mechanism. The mechanism would pair inputs with the desired output responses and adjust the connection strengths accordingly. Such a process involving more than a few layers, however, appears biologically implausible, and its performance may scale poorly as the number of layers is increased.

In a complex network, or in an animal's brain, it is totally unclear how a component layer is to "decide" what transformation its connections should perform—if we assume that the layer needs to "know" what environmental features are important for the animal to respond to. This is

the classic artificial intelligence credit assignment problem: if the final output from a complex system is correct, which connections should be rewarded or strengthened?

The approach we propose avoids this problem. Instead of requiring that a connection or layer "know about" the ultimate goals of the animal, we use only local information. The information that reaches a layer is processed so that the maximum amount of information is preserved. We have seen that this does not in general lead to a trivial one-to-one identity mapping, in which each M cell receives input from only one L cell. In general, the identity mapping is not a solution that maximally preserves information, owing to the role of noise in our model. Instead, each M cell tends to respond to features that are statistically and information-theoretically most significant, in a sense similar to that of principal component analysis. Applying the principle of maximum information preservation to each layer of processing in turn, results in the emergence of a sequence of feature-analyzing functions.

The following analogy may help you to see intuitively how the process works. Imagine a person in an organization, whose job is to make the most informative possible summary of the data that he receives each week. The type of data he receives depends upon the environment external to the organization, the structure of the organization (what "layer" he is part of), and various constraints. Over time, he finds that a particular representation of information—for example, graphical plots involving various variables—serves him best in preparing his summary. If he is allowed to interact with others in his "layer", the criterion can be broadened (as we did for the cells) to state that the composite output of his layer should be as informative as possible.

Note that some set of processing functions will end up being provided by this person's "layer", without the workers needing to know either what the goals of the entire organization are, or what information is deemed most important by their superiors in later "layers".

In both the organizational analogy and the real network, there is no need for any higher layer to attempt to reconstruct the raw data from the summary. The point is rather to enable the higher layers to use environmental information to discriminate the relative value of different actions. If the needed information has been lost at intermediate stages, it cannot

be used. If a local optimization principle is to be used—one that does not attempt to take account of remote high-level goals—then we do not know what particular information is going to be needed at high levels. Since we don't know what information we can afford to discard, it is reasonable to preserve as much information as possible within the imposed constraints. The principle of maximum information preservation thus appears to be an extremely natural and attractive one to use in the construction of a layered perceptual system.

Evolution and infomax. The infomax principle may determine what transformation each layer of a given network will implement. However, it does not specify the "gross architecture" of the network; that is, which layers provide input to which other layers. Nor does it specify the various parameters that may affect layer development, such as noise level, the allowed range of lateral connections, and so on. These aspects of the design may be determined by biological evolution, or by other principles not yet identified.

For an analogy, think of an electronic circuit designer who is not free to modify the properties of the components he or she uses, but who can connect them to form a variety of circuits. In the case of our proposed principle, each "component" is an entire cell layer, and the infomax principle determines that layer's behavior given a particular gross architecture or "circuit design". Thus evolution can "close the loop" on the design process, favoring the survival of organisms whose perceptual systems are well-adapted to their environment.

There is a separate and important evolutionary function that a *generic* principle for the development of a perceptual network layer—whether it be infomax or some other principle—can serve. Suppose that an evolutionary mutation produces a modified eye, or merges auditory signals into the visual pathway at some new point. If there were no generic principle for layer development, we might imagine that mutations would have to occur simultaneously in the processing function of several layers, for those layers to be able to use the novel input properly. But if there is such a generic principle—one that applies to each layer regardless of what type of input reaches it—then the novel input will automatically be processed in accordance with that principle. This suggests that the existence of a generic principle may greatly

increase the likelihood of a mutation being adaptive.

A broader context. Other complex systems, besides neural networks, pose challenges similar to those we have discussed. How might complex structures and behaviors that may appear goal-oriented emerge from relatively simple local rules? We have seen that a local dynamical rule of Hebb type, acting at synapses, leads to an optimization principle—variance maximization—at the level of the whole cell. This suggested an optimization principle—maximum information preservation—that may apply at the level of an entire layer. From the standpoint of information theory, we may find that the immune response system and biological evolution, among other complex systems, have certain abstract similarities to the process of neural development and plasticity, although the dynamical rules and the substrates upon which they act are quite different.

A great deal of work remains to be done, if we are to take this or some other proposed organizing principle, extract testable predictions from

it, and determine its scope and limitations. We need to identify and test such principles, in order to complement and help to focus the enormous amount of detail being revealed by progress in experimental neuroscience. The study of such principles may also provide the understanding needed to develop synthetic perceptual systems that require no explicit programming. □

References

1. R. Linsker, "From Basic Network Principles to Neural Architecture" (series), *Proc. Nat'l Academy of Sciences USA*, Vol. 83, Oct.-Nov. 1986, pp. 7508-7512, 8390-8394, 8779-8783.
2. D.O. Hebb, *The Organization of Behavior*, Wiley, New York, 1949.
3. D.H. Hubel and T.N. Wiesel, "Brain Mechanisms of Vision," *Scientific American*, Vol. 241, Sept. 1979, pp. 150-162.
4. D.H. Hubel and T.N. Wiesel, "Functional Architecture of Macaque Monkey Visual Cortex" (Ferrier lecture), *Proc. Royal Society London*, Vol. B198, 1977, pp. 1-59.
5. J.J. Hopfield, "Neural Networks and Physical Systems with Emergent Collective Computational Abilities," *Proc. Nat'l. Acad. Sci. USA*, Vol. 79, April 1982, pp. 2554-2558.
6. E. Oja, "A Simplified Neuron Model as a Principal Component Analyzer," *J. Math. Biology*, Vol. 15, 1982, pp. 267-273.
7. P. Huber, "Projection Pursuit," *Ann. Statistics*, Vol. 13, No. 2, June 1985, pp. 435-475.
8. P.B. Liebelt, *An Introduction to Optimal Estimation*, Addison-Wesley, Reading, Mass., 1967.
9. R. Linsker, "Development of Feature-Analyzing Cells and their Columnar Organization in a Layered Self-Adaptive Network," in R. Cotterill, ed., *Computer Simulation in Brain Science*, Cambridge Univ. Press, pp. 416-431, in press.
10. B. Widrow and S.D. Stearns, *Adaptive Signal Processing*, Prentice-Hall, Englewood Cliffs, N.J., 1985.
11. C.E. Shannon, "A Mathematical Theory of Communication," *Bell Systems Tech. J.*, Vol. 27, 1948, pp. 623-656.
12. R. Linsker, "Towards an Organizing Principle for a Layered Perceptual Network," in D. Anderson, ed., *Neural Information Processing Systems—Natural and Synthetic*, Amer. Inst. of Physics (NY), to appear.
13. T. Kohonen, *Self-Organization and Associative Memory*, Springer-Verlag, 1984.



Ralph Linsker is on the research staff of the IBM T.J. Watson Research Center. His chief research interests are in the fields of neuroscience and machine perception and learning. Since coming to IBM in 1981 his interests have also included computer design and design automation and medical laser applications. He holds five patents and received IBM's Outstanding Innovation Award for a printed-circuit interconnection design system that has been in production use at IBM since 1983. In the field of medical lasers, he and his colleagues found that pulsed far-ultraviolet radiation can ablate blood-vessel lesions without thermal damage.

Linsker received the BA (summa cum laude) and PhD (in 1972) in theoretical physics from Columbia University, and the MD (in 1976) from Cornell University Medical College. He has done work in theoretical physics at the Princeton Plasma Physics Laboratory, and medical work at New York Hospital.

Linsker's address is Rm. 35-110, IBM T.J. Watson Research Center, PO Box 218, Yorktown Heights, NY 10598.

NeuralWare's Introduction to Neural Computing

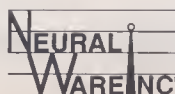
FREE

Teach your PC to learn with neural computing. Expert system generation, forecasting, noise filtering and process control are just a few of the many exciting and innovative industrial applications for neural computing.

Now you can learn more about neural computing, its history and its applications from the author of the *1987 Annotated Bibliography of Neuro-Computing* and the founder of the industry's largest selling neural computing software company, Casimir C. "Casey" Klimasauskas.

For your free booklet, "Teaching Computers to Learn: Applications for Neural Computing" and for information on our neural computing products, send in the coupon below:

Name _____
 Company _____
 Address _____
 City _____ State _____ Zip _____
 Phone _____



103 Buckskin Court,
 Sewickley, PA 15143
 (412) 741-5959

ADVANCE ANNOUNCEMENT

The theme for this year's symposium is "Integrated Networking and Open Architecture." The symposium will highlight topics related to the design, modeling, protocols, performance, and implementation of current and soon to be available network systems.

Three tutorial sessions are offered on April 11th to provide in-depth instruction in the following areas:

- Network Security
- MAP/TOP/GOSIP—Open Systems for the Real World
- The Upper Three Layers—Putting It All Together

Papers dealing with the following topics will be presented on April 12th:

- | | |
|---|---|
| ■ Analysis and Optimization in Distributed Networks | ■ Network Security |
| ■ OSI Protocols | ■ Congestion Control and Load Balancing |
| ■ Issues in Multi Access System | ■ Protocol Specification and Verification |

Papers dealing with the following topics will be presented on April 13th:

- | | |
|--|--------------------------------|
| ■ Resource Allocation in Integrated Networks | ■ Distributed System |
| ■ Protocol Implementations | ■ Network Management |
| ■ Routing | ■ Integrated Networking Issues |

The symposium will also include a panel discussion entitled "National Data Network."

computer networking symposium

April 11-13, 1988 ■ Sheraton National Hotel ■ Washington, DC Area

REGISTER TODAY!! For additional information, please contact: Conference Department, The Computer Society of IEEE, 1730 Massachusetts Avenue, N.W., Washington, DC 20036, (202) 371-1013.



THE COMPUTER SOCIETY
OF THE IEEE



THE INSTITUTE OF ELECTRICAL
AND ELECTRONICS ENGINEERS, INC.

IEEE

STANDARDS

Editor: Helen M. Wood, National Bureau of Standards, B154 Technology, Gaithersburg, MD 20899; (301) 975-3240

Project launched for application layer service definition

Accredited Standards Committee X3 has approved work on a new project to develop an American national standard for application layer service definition supported by presentation connectionless mode transmission. The actual development work will be carried out by Technical Committee X3T5, Open Systems Interconnection (OSI).

The goals of X3T5 are to contribute to the international effort on the definition of application layer service supported by presentation connectionless

mode transmission, and to adopt an American national standard upon acceptance as an international standard.

This work will be limited to the definition of the application service provided by the application layer of the OSI reference model supported by presentation connectionless mode transmission. It is also applicable to systems requiring a connectionless mode of operation to satisfy certain specific application requirements.

Since X3T5 intends to complete this

draft American national standard upon its acceptance as an international standard, interested participants and users are encouraged to become involved in the process as early as possible to influence the international work that will be undertaken later.

To join X3T5 to work on this project or any other project under way, contact the X3T5 chair, Jerrold S. Foley, Electronic Data Systems Corp., 300 E. Big Beaver Rd., PO Box 7019, CUBE 5176, Troy, MI 48083; (313) 524-8416.

NBS completes first part of Defense Communications Agency project

Researchers at the National Bureau of Standards have completed the first part of a project to build gateways between US Defense Dept. communication protocols and Open Systems Interconnection (OSI) networking protocols. The project was sponsored by the US Defense Communications Agency (DCA).

OSI standards enable the equipment and systems of different manufacturers to communicate with each other through networks. The gateways will be used to maintain the agency's operations during

conversion to OSI networks.

Working with a guest scientist from IBM, the researchers in the NBS Institute for Computer Sciences and Technology have developed gateways and a test system for linking the Defense Dept. electronic mail protocol (Simple Mail Transfer Protocol) to its OSI counterpart (Message Handling Facility/X.400). The automated test system is used to determine whether products comply with the gateway specifications.

Specifications are still being developed for a second gateway to connect

the military's File Transfer Protocol and OSI's File Transfer, Access, and Management Protocol. NBS expects to complete work on the second gateway, including a test system, by the end of this year.

When completed, these specifications and testing procedures will be available to computer and communications vendors as well as to test service organizations for use in developing gateway products that link Defense Dept. and OSI protocols.

Standards for DES encryption on 802.3 LANs to be discussed at DC meeting

A meeting will be conducted March 24 in Washington, DC, to discuss the standardization of DES encryption on 802.3 LANs. Sponsored by the IEEE Technical Committee on Security and Privacy, the meeting will be held a day after completion of Compstan 88, the three-day 1988 Conference on Computer Standards.

All those in the industry interested in discussing and participating in the development of this standard are encouraged to attend.

Several vendors are developing products to provide secure communications on 802.3 LANs. The products are similar, although there are enough differences that a secure multi-vendor 802.3 LAN is not possible. As vendors know, customers buy from many sources and expect to effect communications among the various products. All customers need security, from the most

stringent government requirements for secrecy to the less stringent, but equally important, commercial requirements for privacy. Thus, using standard security mechanisms would be advantageous in marketing any LAN product.

The primary discussion at the meeting will focus on determining if a standard is necessary and defining the scope of the standard activities. Two areas will be prepared for discussion: encryption and key management.

Topics to be addressed in the area of encryption will involve the need for a common encryption algorithm (DES is a prime candidate); the standardization of the mode of operation (that is, block, cyphertext feedback); the establishment of a common key length for the encryption algorithm and the public key; and the definition of which fields should be encrypted.

Topics to be addressed in relation to

key management relate to the need for a common key management scheme, and provisions for broadcast and multicast packets.

Discussion about the key management scheme will span the use of schemes that are centralized, distributed, or a combination of both. A distributed system would most likely employ a public key system for creating session keys without involving a third party in establishing the connection. A centralized system would require an online key management system (third party) to monitor and authorize all connection attempts across the network. A hybrid system containing both distributed and centralized key management concepts will also be considered.

For more information on this activity, contact Kim Kirkpatrick at (617) 271-7555.

IEEE Board approves Computer Society standards activities

The IEEE Standards Board approved the following new Computer Society projects at its December 1987 meeting:

- P1003.0, Guide to POSIX Based Open System Architecture
- P1003.5, Ada Language Binding for POSIX
- P1003.6, Security Interface Standards for POSIX
- P1016.2, Guide to Software Design Descriptions
- P1163, A Standard Interface for IEEE Standard 1076-1987 to Computer Aided Design and Manufacturing (CAD/CAM) tools

- P1164, Recommended Practice for Coding in IEEE Std. 1076-1987
- P1165, Recommended Practice for the Interrelationships Between IEEE 1076 VHDL and EIA RS44 EDIF
- P1173, Standard Set of Functional Components of a Simulation Language

The board also approved six new standards:

- 802.3D, LAN: Fiber Optic Inter-Repeater Link
- 961, Eight Bit Micro Bus System

- 1000, Eight Bit Backplane Interface
- 1058.1, Software Project Management Plan
- 1063, Software User Documentation
- 1076, Description Language for Electronic Hardware

For more information on current IEEE projects, or assistance in purchasing copies of draft or final standards, contact the IEEE Computer Standards Secretariat, 1730 Massachusetts Ave. NW, Washington, DC 20036-1903; (202) 371-0101.

MIT, leading workstation vendors form consortium to support windowing standard

The Massachusetts Institute of Technology and several leading computer companies have formed the MIT X Consortium. The objective is to support and further develop the X Window System, an industry standard graphics windowing system for workstations in multitasking, networked computing environments.

The system has been adopted by software and hardware developers as an industry standard method for simultaneously displaying multiple applications on a computer screen, according to MIT. With it, workstation users in a multivendor network can simultaneously access a wide array of computing resources, such as supercomputers or specialized processors, through win-

dows on the workstation screen.

The consortium, open to organizations dedicated to promoting the evolution of the X standard, is responsible for maintaining the documentation of the major components of the X system and a software validation suite for server implementations and client interfaces. It will also fund systems enhancements, such as 3D graphic extensions and libraries, video extensions, and protocol binding, and toolkits for languages other than C.

The founding consortium members include Apollo Computer, Apple Computer, AT&T, CalComp, Digital Equipment Corp., Hewlett-Packard, Sequent Computer Systems, Sony, Sun Microsystems, Tektronix, and Xerox.

Text processing, office, and publishing systems group seeks participants

The X3V1 technical committee on text processing, office, and publishing systems standards is working on protocols for distributed office applications and other text interchange activities. Persons with expertise or general interest in this area are invited to attend the committee's meetings.

Distributed office applications are characterized by the availability of a diverse array of electronic office equipment and enabling software, dispersed geographically but capable of acting cooperatively by means of communications media and techniques. Facilitating the homogeneous interconnection between these heterogeneous offerings

is extremely important to realize their full benefits.

X3V1 is currently concentrating on developing a general model for distributed office application protocols, standards for print service access, and document filing and retrieval.

For further information, contact the chair of the distributed office applications task group, Robert Christie, Control Data Corp., 4201 N. Lexington Ave., St. Paul, MN 55126-6198; (612) 482-6689.

For more information on the X3V1 committee, contact its chair, L.M. Collins, IBM Corp., 200 Las Colinas Blvd., Irving, TX 75039; (214) 556-4390.

Standards Briefs

Posix goes international

The joint ISO/IEC international standards committee, known as JTC1, has approved the establishment of an international working group on POSIX.

Headed by the Computer Society's Jim Isaak, chair of the P1003 POSIX Subcommittee, this new working group will coordinate international review and comment on the IEEE Trial Use Standard for Portable Operating Systems Environments (POSIX).

For information on POSIX standardization activities, contact the IEEE Computer Standards Secretariat, 1730 Massachusetts Ave. NW, Washington, DC 20036-1903; (202) 371-0101.

IEEE Standards Office slates computer software seminars in San Diego, London

The IEEE Standards Office will conduct a series of computer software seminars in April, May, and June in San Diego, Calif., and London, England.

"Software Verification & Validation" and "Software Quality Assurance" will be offered April 18-20 in San Diego and May 30-June 1 in London.

"Software Testing" and "Software Configuration Management" will be offered April 21-22 in San Diego and June 2-3 in London.

In addition, "Software Requirement Specification" will be offered in San Diego April 21-22.

For additional information on the seminars, contact Seminar Manager, Standards Office, IEEE Headquarters, 345 East 47th St., New York, NY 10017-2369; (212) 705-7759.

Call for Papers, Participants, and Exhibitors

FRONTIERS '88

The 2nd Symposium on the

Frontiers of Massively Parallel Computation

October 10-12, 1988

George Mason University
Fairfax, Virginia

Programming Languages
Architectures
Algorithm Development
Graph Theory
Image Processing
Numerical Modeling
Data Base Management
Interconnection Networks
Hierarchical Structures
Neural Networks
New Technologies

IMPORTANT DATES

Abstracts due:	April 15, 1988
Acceptances sent:	June 15, 1988
Registrations due:	September 15, 1988
Tutorials:	October 10, 1988
Sessions and Exhibits:	October 11-12, 1988
Manuscripts due:	October 12, 1988

CONTACT INFORMATION

To receive an advance program with registration information, send name, address, and phone number to:
Frontiers '88 Symposium
P.O. Box 334
Greenbelt, Maryland 20770-0334

Submit abstracts to:
Prof. David H. Schaefer
Department of Electrical and Computer Engineering
George Mason University
4400 University Drive
Fairfax, Virginia 22030

To present an exhibit:
Mr. James R. Flesher
Chairman, Frontiers '88
Image Analysis Facility, Code 635
NASA/Goddard Space Flight Center
Greenbelt, Maryland 20771
(301) 286-3464

BACKGROUND AND SUBMISSION INFORMATION

The Frontiers '88 Symposium will focus on the increasing importance of massively parallel computer systems and data parallel programming techniques. Applications, theory, and architectures for computers with more than 1,000 processors will be discussed by users, scientists, and developers from industry, academia, and Government. Selected papers will be published in a dedicated issue of the *Journal of Parallel and Distributed Computing*. Tutorials will survey the concepts embodied in massively parallel systems and broad applications areas. Exhibits will feature commercial and prototype massively parallel systems, relevant journals, and newsletters.

Those interested in presenting a paper must submit a three-page abstract to the program chairman no later than April 15, 1988. Complete names, addresses and telephone numbers for all authors (beginning with primary author) must accompany the abstract. Letters of acceptance and guidelines for preparing a final paper will be sent to authors by June 15, 1988. Authors who prefer to present their paper as part of a poster session should so indicate. The program chairman reserves the right to assign authors to a poster session in the event of greater than expected response.

PROGRAM COMMITTEE

Prof. David Schaefer, Chairman
George Mason University

Dr. Ray Arnold
NASA Headquarters

Dr. Ken Batchner
LORAL Defense Systems-Akron

Dr. Jack Dongarra
Argonne National Laboratory

Prof. Michael Duff
University College, London

Dr. Milton Halem
NASA/Goddard Space Flight Center

Dr. James Hardy
Whitney/Demos Productions

Prof. Dennis Parkinson
Active Memory Technology

Prof. Tomaso Poggio
Massachusetts Institute of Technology

Prof. John Reif
Duke University

Prof. Anthony Reeves
University of Illinois

Prof. Azriel Rosenfeld
University of Maryland

Dr. Paul Schneck
Supercomputing Research Center

Dr. Steven Squires
DARPA

Dr. Guy Steele
Thinking Machines Corporation

Prof. Leonord Uhr
University of Wisconsin

Sponsored by George Mason University,  Computer Society of the IEEE (pending),  IEEE-National Capital Area Council, and NASA

Supported by grants from Active Memory Technology, LORAL Defense Systems-Akron, Martin Marietta Aerospace, Science Applications Research, and Thinking Machines Corporation

Editor: Sallie Sheppard, Office of Associate Provost for Honors Program and Undergraduate Studies, Texas A&M University, College Station, TX 77843; (409) 845-3210

Society's handbook distributed to members

A handbook describing the varied professional resources and benefits of Computer Society membership has been mailed to the 90,000 members of the society. The pamphlet was designed to serve as a reference to the services and programs available to current members as well as to potential members.

The handbook describes the society's organization and offices; periodicals (like *Computer*) and Computer Society Press offerings; the conference, awards and ombudsman programs; chapter, educational, standards, and student activities; the purpose and scope of each of the society's 33 active technical committees plus the Committee on Public Policy; electronic mail services; and the various ways participation in the society can help members enjoy and enhance their careers.

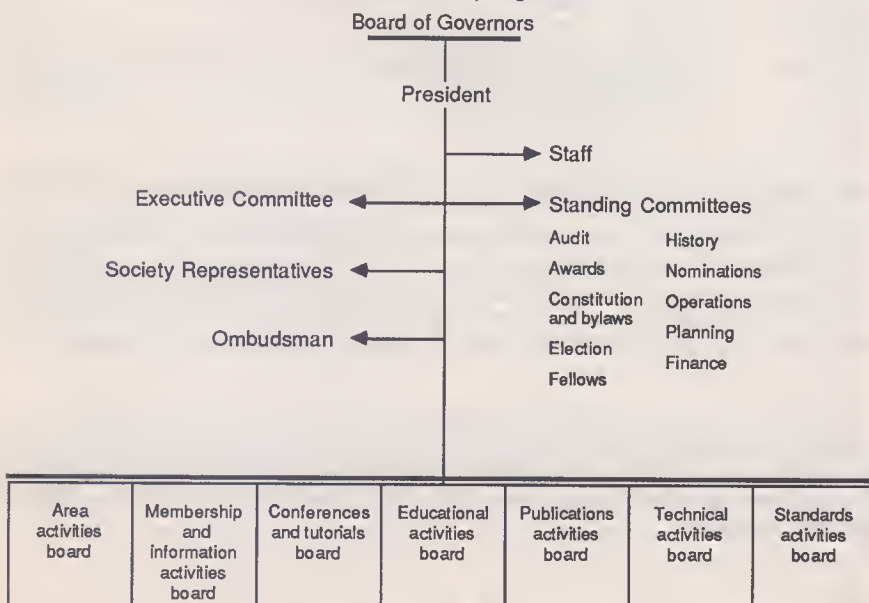
The booklet outlines the history of the society and the Institute of Electrical and Electronics Engineers, tracing their origins to 1884 when the American Institute of Electrical and Electronics Engineers was formed. In 1963, the

AIEE merged with the Institute of Radio Engineers (which itself had been in existence for half a century) and the new organization adopted the IEEE as its name. With some 300,000 current members, the IEEE has become the world's largest technical professional organization in terms of scope and size.

The Computer Society, the largest of the IEEE's 33 technical units, evolved with formation of the IEEE from computer groups within the two parent organizations. From these beginnings, the CS has grown to include a large and diverse membership united by a common interest: advancing the theory and practice of computer science and engineering. The organization brings together elected officials, volunteers, and society staff to serve the membership in promoting cooperation and the exchange of technical information.

If you did not receive your copy of the handbook or want to obtain additional copies for yourself or colleagues, circle 188 on the Reader Service Card.

The Computer Society Organization



1988 TC chairs

Computational Medicine

John Long
University of Minnesota
Suite 408
2829 University Ave. SE
Minneapolis, MN 55414
Off.: (612) 627-4850
Res.: (612) 942-0140
Comppmail + j.long

Computer Architecture

Roger Anderson
Lawrence Livermore Laboratory
L-306
PO Box 808
Livermore, CA 94550
Off.: (415) 422-4239

Computer Communications

William Livingston
Vance Systems
3901U Bonanza Blvd.
Chantilly, VA 22021
Off.: (703) 471-9402
Res.: (703) 624-3326
Comppmail + w.livingston

Computer Elements

Ronald Bell
Sperry
322 N. Sperry Way
Salt Lake City, UT 84116
Off.: (801) 594-5386

Computer Graphics

Lawrence Rosenblum
Naval Research Laboratory
Code 5810
4555 Overlook Ave. SW
Washington, DC 20375
Off.: (202) 767-3743
Res.: (301) 424-5762
Comppmail + l.rosenblum

Computer Languages

Pei Hsia
University of Texas/Arlington
Computer Science
2100 Oak Bluff Dr.
Arlington, TX 76001
Off.: (817) 273-3785
Res.: (817) 275-8764
Comppmail + p.hsia

Computer Packaging

Martin Freedman
IBM T.J. Watson Research Center
13-209
PO Box 218
Yorktown Heights, NY 10598
Off.: (914) 945-1711
Res.: (203) 322-9570

Computers in Education

Doris Carey
University of Nevada
4505 Maryland Parkway
Las Vegas, NV 89154
Off.: (702) 739-3860
Res.: (702) 641-7090
Comppmail + d.carey

Computing and the Handicapped

Elmer Hoyer
Wichita State University
Box 44
Dept. of Electrical Engineering
Wichita, KS 67208
Off.: (316) 689-3415
Res.: (316) 744-1929

Database Engineering

Sushil Jajodia
National Science Foundation
Rm. 310
Washington, DC 20550
Off.: (202) 357-9572
Res.: (703) 243-0471
Compmail + s.jajodia

Design Automation

Sumit DasGupta
IBM, East Fishkill Facility
Zip 3A1
Route 52
Hopewell Junction, NY 12533
Off.: (914) 894-0540
Res.: (914) 297-1946
Compmail + s.dasgupta

Display Ergonomics

James Greeson, Jr.
IBM
H-29-061
PO Box 12195
Research Triangle Pk., NC 27709
Off.: (919) 543-6655
Res.: (919) 787-3910
Compmail + j.greeson

Distributed Processing

Walter Kohler
University of Massachusetts
Dept. of ECE, Marcus Hall
Amherst, MA 01003
Off.: (413) 545-0765
Compmail + w.kohler

Fault-Tolerant Computing

David Rennels
Computer Science Dept.
4731 Boelter Hall
University of California
Los Angeles, CA 90024
Off.: (213) 825-4033
cs.ucla.arpa

Mass Storage Systems

Patric Savage
Shell Development
PO Box 481
Houston, TX 77001
Off.: (713) 663-2384
Res.: (713) 934-8063
Compmail + p.savage

Mathematical Foundations of Computing

Ashok Chandra
IBM T.J. Watson Research Center
Route 134
PO Box 218
Yorktown Heights, NY 10598
Off.: (914) 945-1752
Res.: (914) 739-2175
Compmail + a.chandra

Microprocessors/Microcomputers

Martin Freeman
Stanford University
Center for Integrated Systems
Stanford, CA 94305
Off.: (415) 991-3591
Res.: (415) 493-5382
Compmail + m.freeman

Microprogramming

Joseph Linn
Institute for Defense Analysis
Computer/Software Engineering
1801 N. Beauregard St.
Alexandria, VA 22311
Off.: (703) 824-5500
Res.: (703) 671-8527
Compmail + j.linn

Multiple-Valued Logic

Michel Israel
IEEE-CNAM
18, Allee Jean-Rostand
Evry Cedex, France
Phone: (1) 6-077-9740

Oceanic Engineering

Michael Guberek
Global Imaging
201 Lomas Santa Fe Dr.
Suite 360
Solana Beach, CA 92075
Compmail + m.guberek

Office Automation

David Choy
IBM Almaden Research Laboratory
MS K52/803
650 Harry Rd.
San Jose, CA 95120
Off.: (408) 927-1846
Res.: (415) 969-1811
Compmail + d.choy

Operating Systems

Joseph Boykin
Encore Computer
257 Cedar Hill St.
Marlboro, MA 01752-3004
Off.: (617) 460-0500
Res.: (617) 845-1074
Compmail + j.boykin

Optical Processing

Ravi Athale
BDM
7915 Jones Branch Dr.
McLean, VA 22102
Off.: (703) 848-7556

Pattern Analysis and Machine Intelligence

J.K. Aggarwal
University of Texas
Electrical Engineering Dept.
Austin, TX 78712
Off.: (512) 471-3259
Res.: (512) 451-4697
Compmail + j.aggarwal

Personal Computing

Walter Beam
Systems Engineering Dept.

George Mason University

Fairfax, VA 22030
Off.: (703) 323-2782
Res.: (703) 370-3431
Compmail + w.beam

Real-Time Systems

Andre van Tilborg
2600 Riggs Way Parkway
Minneapolis, MN 55413

Robotics

Mohan Trivedi
University of Tennessee
Ferris Hall
Dept. of Electrical and Computer
Engineering
Knoxville, TN 37996
Off.: (615) 974-5450
Res.: (615) 523-5853
Compmail + m.trivedi

Security and Privacy

Carl Landwehr
Naval Research Laboratory
Code 5593
Washington, DC 20375
Off.: (202) 767-3381

Simulation

Heimo Adelsberger
Vienna Business School
Augusse, 2
A-1090 Vienna, Austria
Res. phone: (43) 222-476-227

Software Engineering

Lorraine Duvall
Duvall Computer Technologies
PO Box 568
Rome, NY 13440
Off.: (315) 337-1564
Res.: (315) 339-2592
Compmail + l.duvall

Supercomputing

Joanne Martin
IBM
12G-05
44 South Broadway
White Plains, NY 10601
Off.: (914) 789-7508
Res.: (914) 686-6284
Compmail + j.martin

Test Technology

Paul Bardell
IBM
Dept. 256, Box 003
PO Box 950
Poughkeepsie, NY 12602
Off.: (914) 433-5400
Compmail + p.bardell

Very Large Scale Integration

Don Bouldin
University of Tennessee
420 Ferris Hall
Dept. of Electrical and Computer
Engineering
Knoxville, TN 37996-2100
Off.: (615) 974-5444
Res.: (615) 966-8527

Classification of 'sensitive' information is once again in civilian hands

Ralph J. Preiss, COPP Chair

On January 8, 1988, President Reagan signed into law the Computer Security Act of 1987 (P.L. 100-235). According to the Congressional Record, the major provision of the act makes the National Bureau of Standards the primary agency of the government "responsible for sensitive civil sector computer matters."

The law provides for the development of standards and guidelines for the security of federal computer systems and establishes a training program (in computer data security) for all persons involved in federal computer systems.

It states that the National Security Agency will continue to have authority over classified information in federal computers, and it directs the two agencies to work together to avoid duplication of effort or the promulgation of conflicting guidelines in the process. The US Senate passed the act on December 21, 1987 based on HR 145, which the House of Representatives passed on June 22, 1987.

Computer Society members will recall that, in the fall of 1984, President Reagan issued National Security Decision Directive (NSDD) 145 giving the US Department of Defense the authority to classify certain "sensitive" information in unclassified files. The objective was

IEEE/USAB selects Fernbach for Citation

Sidney Fernbach, consultant for Control Data Corp. and active Computer Society volunteer, has been selected to receive the Citation of Honor for 1987 presented by the IEEE's United States Activities Board.

The Committee on Communications and Information Policy (CCIP) nominated Fernbach for his contributions on improving and maintaining the strength of US supercomputer capability. Fernbach has led the IEEE supercomputer effort for the past six years, first as chair of the Ad Hoc Committee on Scientific Supercomputers, and then as chair of the CCIP Subcommittee on Scientific Supercomputers.

Fernbach, Gerald W. Gordon, F. Karl Willenbrock, and James A. Watson will receive plaques at upcoming meetings of the USAB and the Professional Activities Council for Engineers (PACE).

to insure that this information was inaccessible to non-US citizens.

According to the New York Times, the administration's main concern was that the broad range of technical and scientific information available in government and commercial computers would be readily available to our "adversaries" unless safeguards were taken. The Times reported that Donald Latham, an assistant secretary of the DoD, had testified in 1985 that "virtually every aspect of government and private information is readily available" (in computer-readable and analyzable form)...and that "unfriendly governments and international terrorist organizations are finding easy pickings" from the flood of unprotected information.

One of the first victims of this directive was the International Test Conference, whose organizers were told in no uncertain terms that a number of papers had to be removed from the published proceedings because they contained what was considered to be sensitive information.

COPP members have cooperated with the ITC, the DoD, and the IEEE USAB to resolve the published-proceedings matter. COPP has also helped formulate an IEEE position on sensitive information. The Computer Society's Board of Governors didn't accept the position but, instead, passed a resolution urging the president of the IEEE to inform the DoD that we do not recognize "sensitive" as a third and arbitrary classification. A compromise became necessary when the military showed "sensitive information" to be analogous to the way civilians label "company proprietary information," except that NSDD 145 labels it as "US proprietary."

The best IEEE could do was to urge transferring the labeling responsibilities back to civilian control. Thus, the IEEE USAB's Committee on Communications and Information Policy (CCIP) worked with Congress (the chair testified twice in 1987) and reviewed the definitions in HR 145 before Congress voted the measure into law.

This action supersedes the President's NSDD 145 and returns the control of information classification rules to civilian hands after three years under sole DoD discretion.

Report examines recession's effects on computer industry

Concern about the effects of a possible recession and a slowdown in capital spending on the computer industry is exaggerated, according to a forecast by Sanford C. Bernstein & Co.

The company's technology research group states that the correlation widely assumed to exist between total capital spending and computer industry revenues is almost nonexistent and that capital spending and computer revenues have moved in opposite directions in most years.

The forecast cites product cycle as the major determinant of computer industry revenue growth. Strong product cycle years tend to show strong revenue growth regardless of economic conditions, and years between product cycles show slow growth even during economic expansion.

The Bernstein technology group has also projected the impact of a revenue slowdown on individual computer companies to demonstrate that the effect on earnings would vary widely from company to company.

Overall, Bernstein & Co. sees the earnings of major companies growing at 25 to 50 percent in 1988 on the basis of major new product cycles. For companies between product cycles, the estimated growth rate is 15 to 25 percent.

Article looks at means for verifying INF treaty

The February issue of *IEEE Spectrum* includes an article by Associate Editor John A. Adam in which he considers the technological means for verifying the Intermediate-Range Nuclear Forces Treaty signed late last year.

The article, entitled "Verifying the New Arms Pact," examines provisions of the INF agreement, possible enforcement problems, the effect of the agreement on this summer's Strategic Arms Reduction Talks, the impact of Soviet visits to US weapons sites, and plans for destroying weaponry.

UPDATE

Contributions to Update are welcome. Send news of industrial or university research and of public policy or professional issues to Update Editor, 10662 Los Vaqueros Circle, Los Alamitos, CA 90720, or to Bruce D. Shriver, Editor-in-Chief, IBM T.J. Watson Research Center, PO Box 704, HO-B04A, Yorktown Heights, NY 10598.

Starting salaries in data processing at record highs

Starting salaries for data processors have reached an all-time high in 1988, according to a study conducted by Robert Half International, data processing and financial recruiting specialists. The average increase over 1987 is 4.2 percent.

Some specific starting-salary ranges

are \$37,000-\$44,000 for project managers at medium-size installations, a 5.2-percent increase; \$61,000-\$81,000 for management information systems directors at large installations, a 1.4-percent increase; \$20,000-\$25,000 for programmers at small installations, a 5.9-percent increase; and \$36,000-\$44,000

for systems analysts at large installations, a 2.6-percent increase.

These are national averages, and geographic variances should be applied to all data processing starting salaries below \$50,000, according to Robert Half International.

Questions raised about SDI National Test Bed

The US Strategic Defense Initiative Organization has awarded the contract for the National Test Bed to Martin Marietta Corp. The \$500-million contract is for five years to develop a national network of supercomputer and simulation facilities designed to evaluate the feasibility of the Strategic Defense Initiative, commonly called "Star Wars." With an estimated completion cost of \$1 billion, the project would be the largest single item in the SDI budget.

Members of Computer Professionals for Social Responsibility, a nonprofit public interest organization of people in the computing field, have studied the project's official request for proposals, as well as available news stories. The organization's preliminary analysis resulted in the following conclusions:

- The feasibility of SDI cannot be determined from computer simulation. Consequently, the National Test Bed will not establish whether SDI will work as intended.

- The accuracy and adequacy of computer simulations depend on the validity of the model of events and system properties used. With SDI, there can be no validation of the model because too many relevant variables are unknown and will remain unknown.

- Dept. of Defense officials have maintained that the US should continue to test nuclear warheads for assurance that they are reliable. These officials have repeatedly stated that computer

simulation of warhead design and performance is not an adequate substitute for actual explosive testing of real warheads.

- Personnel managing the National Test Bed will become a built-in constituency pressuring policymakers to abandon the constraints of the ABM treaty. NTB managers will undoubtedly argue that their simulations would benefit from data from real space-based component testing, including testing that would violate the treaty. But even component testing linked to computer simulation will not assure us that SDI will work right the first time it is called into action as a complete system.

- The Reagan administration has portrayed SDI as a research project. The National Test Bed is a *development* project. Rather than being limited to performing computer simulations of ballistic missile defense, it will be a prototype of an immense command-and-control system for battle management in case of nuclear war.

- SDI cannot be made reliable, because it is impossible to test the system under realistic conditions of actual use; furthermore, the system's performance characteristics cannot be known well enough in advance to specify the required detail and accuracy in computer software.

In summary, the organization, based in Palo Alto, Calif., feels that "the National Test Bed is a waste of taxpayers' money."

Neural networks may someday detect forgers

Joseph Goodman, professor of electrical engineering at Stanford University, and Dorothy Mighell, a graduate student, are pioneering a neural network approach to the problem of credit card fraud.

A neural network can learn by trial and error. During training, the computer is presented with different versions of a signature, because a person's handwriting changes with his or her mood. Once the correct set of weighted features is obtained, the computer will correctly accept or reject a novel version of the signature.

Neural network processing is arranged in layers. As the computer scans the image of a signature, small bits of the image are mapped onto the neurons in the first layer. At each layer, or level, the computer makes a different judgment on the validity of the signature. The signal is processed until it reaches the single processor in the final layer of the network, which is then on or off (accept or reject).

Goodman and Mighell are currently simulating a neural network computer with software. If the approach works, they may build a machine with actual neural network structure.

Other groups at Stanford are also working with neural networks, including several people in the psychology department.

Supercomputer aids study of aircraft control phenomenon

Using NASA's new supercomputer system called the Numerical Aerodynamics Simulation Facility, a researcher has made the most in-depth analysis to date of vortex breakdown, a complex phenomenon that can cause loss of lift and control for high-performance aircraft.

A computer model developed by Kozo Fujii, a research fellow at NASA's Ames Research Center, simulates the air flow field physics associated with vortex breakdown and provides new insights into its causes. Vortex breakdown is difficult to study experimentally and has remained poorly understood.

The Ames computer model is the first computational analysis to predict spiral breakdown, thought to be the most common type of breakdown over aircraft wings. It can also predict bubble breakdown, the other major type. Fujii's model of breakdown also is the first to use a real wing configuration

rather than a simplified form. The model tracks vortex breakdown on strake delta wings such as those found on F-16 and F-18 aircraft. It can be adapted for use with a variety of wing configurations.

The model is three dimensional, time accurate, and highly detailed, with the flow field calculated at 850,000 grid points. It is based on the Navier-Stokes equations, a highly complex set of equations describing how fluids behave, and it requires 25 hours running time on an advanced supercomputer.

Fujii is working to improve the accuracy of the model. At present, the model is valuable for analytical pur-

poses but requires more resolution for design use. One key to achieving higher resolution results, Fujii believes, lies in using an even finer grid in the calculations. However, a finer grid would require costly additional computer time and enormous computer memory—larger than that of existing supercomputers.

A solution is to use a zonal method, which increases the number of grid points in selected areas while keeping computer processing time from increasing. Using this technique, Fujii plans to extend the model to complex geometries, which would include the fuselage as well as wing surfaces.

Research roundtable appoints new members

Five new members have been appointed to serve on the Council of the Government-University-Industry Research Roundtable, a discussion forum of scientists, engineers, administrators, and policymakers sponsored by the National Academies of Sciences and Engineering and the Institute of Medicine.

Newly appointed to three-year terms are Joel S. Birnbaum, vice president and general manager, Information Technology Group, Hewlett-Packard; Richard F. Celeste, governor of Ohio; Kenneth H. Keller, president, University of Minnesota; John E. Sawyer, president emeritus, Andrew W. Mellon Foundation; and Alvin W. Trivelpiece, executive officer, American Assoc. for the Advancement of Science. The new members join 19 current members.

The Research Roundtable, now in its fifth year, was created to foster discussion of crosscutting science and technology policies and problems by high-level representatives of government, universities, and industry. Most of its work is conducted through working groups in special interest areas; the three working groups focus on science and engineering talent, university research and its management, and partnership and joint ventures between government, academia, and private industry.

NSF and NASA to link computer networks

The National Science Foundation has signed an agreement with the National Aeronautics and Space Administration to share high-speed communications lines. The agreement, which will ultimately link university researchers now connected to NSF's national computer communications network to databases and supercomputers at NASA laboratories, is expected to save money by avoiding duplication of efforts by the two agencies.

The agreement is in accord with a report recently released by the White House Office of Science and Technology Policy recommending improvements in networking to enhance US leadership and provide linkages needed for collaborative research by scientists working at different institutions.

Three NASA facilities will be linked

to existing NSF regional networks, which in turn are connected through a national backbone network. The Goddard Space Flight Center in Greenbelt, Md., will be linked to the Southeastern Universities Research Associates Net (SURANet); the Ames Research Center in Mountain View, Calif., will be linked to the Bay Area Regional Research Net (BARRNet); and the Johnson Space Flight Center in Houston will be linked to SESQUINet, a regional network in Texas.

Authorized scientists will be able to remotely access and use NASA data in their research and apply for time on NASA supercomputers. NASA-funded scientists at universities served by NSF regional networks will be able to communicate and collaborate with colleagues at the NASA centers.

Picture darkens for UK computer industry

The British government has rejected proposals for a £1000-million, five-year research program that would have succeeded the Alvey project currently nearing completion. Instead, most research in information technology will be done through the second phase of the European Community's £1100-million Esprit program. The report in the British journal *Nature* states that a modest national initiative has been proposed to complement Esprit.

Under the Alvey program, more than 200 industrial projects received support. A committee had recommended con-

tinuation of the program, with a focus on the application of new technologies and an increased government contribution of £425 million.

The government's response, contained in a white paper, indicates a shift toward more collaborative research and fewer "near-market" initiatives, which the government prefers to have industry pay for.

Nevertheless, the Dept. of Trade and Industry plans to contribute £8 million over three years to a national program of research into high-temperature superconductivity.



25th ACM/IEEE DESIGN AUTOMATION CONFERENCE®

JOIN THE 25th ANNIVERSARY CELEBRATION

For 24 years the Design Automation Conference has been the meeting place for electronic CAD/CAM Engineers

Always 1st in the latest CAD/CAM Technological Innovations

NOW IN THE 25th ANNIVERSARY YEAR, DAC WILL OFFER:



Over 130 papers, tutorials and workshops.



Over 110 vendors of CAD/CAM hardware and software will exhibit their products.



Over 50 vendor-technical presentations on Sunday afternoon June 12th - Many announcing new products.

A MUST FOR
CAD/CAM ENGINEERS
AND DEVELOPERS

DYNAMIC
APPLICATIONS
AND
CONCEPTS

Anaheim Convention Center, Anaheim, CA
June 12-15, 1988

Advance Registration ends May 14, 1988
For More Information, Call 1-303-530-4333



COMPUTER
SOCIETY
OF IEEE®
DATC



THE INSTITUTE OF ELECTRICAL
AND ELECTRONICS ENGINEERS, INC.



HOTEL RESERVATION FORM 25th Design Automation Conference June 12-15, 1988

Hotel reservations will be handled by the Anaheim Visitors & Convention Bureau. All reservations will be handled on a first-come first-served basis. The following hotels are available

RATE					
HOTEL	SINGLE	DOUBLE	HOTEL	SINGLE	DOUBLE
— Convention Center Inn	\$48.00	\$48.00	— Inn at the Park	\$78.00	\$78.00
— Disneyland	\$90.00	\$90.00	— Jolly Roger	\$68.00	\$68.00
— Emerald	\$88.00	\$88.00	— Marriott	\$95.00	\$105.00
— Grand	\$78.00	\$78.00	— Paul Bunyan	\$55.00	\$59.00
— Hilton	\$98.00	\$112.00	— Quality Inn	\$78.00	\$78.00
— Holiday Inn	\$68.00	\$68.00	— Raffles	\$53.00	\$58.00
— Hyatt Regency Alicant	\$82.00	\$85.00	— Sheraton	\$73.00	\$83.00
— Hyatt Anaheim	\$81.00	\$81.00			

PLEASE MAKE FOUR CHOICES AND NUMBER THEM

RESERVATIONS SENT TO THE DAC OFFICE WILL BE DISCARDED

The Anaheim Visitors & Convention Bureau will attempt to make your reservations as indicated by the choice of hotel you specify. If the hotels you choose are not available, the Bureau will make your reservation at an available Conference Hotel. Complimentary busing will be provided from all Conference Hotels listed above the Anaheim Convention Center, when needed

INDICATE HOTEL PREFERENCES ABOVE AND SEND COMPLETED FORM TO:

DAC HOUSING BUREAU c/o ANAHEIM VISITORS & CONVENTION BUREAU
P.O. BOX 4270
ANAHEIM, CALIFORNIA 92803
(714) 999-8939 - FOR INFORMATION ONLY - NO TELEPHONE RESERVATIONS!

FIRST NAME _____ LAST NAME _____
COMPANY _____
MAIL STOP _____
STREET ADDRESS _____
CITY _____ STATE _____ ZIP _____
ARRIVAL DATE _____ TIME _____ AM _____ PM
DEPARTURE DATE _____ TIME _____ AM _____ PM

A deposit of \$75.00 is required by check or credit card. Make check payable to DAC HOUSING BUREAU. Your reservation will be confirmed by both the DAC Housing Bureau and the hotel to which you are assigned

CREDIT CARD INFORMATION

☐ VISA
☐ AMERICAN EXPRESS
☐ MASTERCARD
Number _____
Expiration Date _____

All changes must be made directly with the DAC Housing Bureau listed above, 72 hours before your arrival date or you will lose your deposit

ADVANCE REGISTRATION FORM 25th Design Automation Conference

Save money and avoid conference registration lines by registering today!

IMPORTANT

1. Full payment in U.S. dollars MUST accompany registration. Any registration without payment will be discarded.
2. Advance registrations must be postmarked no later than May 14, 1988
3. Register one person per form.
4. **Refund Policy:** Written requests for refunds must be received in the DAC office no later than May 14, 1988. Refunds are subject to a \$15.00 processing fee.
5. Payment must be made in U.S. dollars and made payable to: 25th DESIGN AUTOMATION CONFERENCE
6. Mail this form or a facsimile to:

Mr. P.D. Pistilli
25th DESIGN AUTOMATION CONFERENCE
7490 CLUBHOUSE RD. #102
BOULDER CO. 80301

Telephone Registrations WILL NOT BE ACCEPTED. NO CREDIT CARDS.
For Information Only, Call (303) 530-4333

Note: Your name and company will be used to print your identification badge.
PLEASE TYPE OR PRINT CLEARLY

FIRST NAME _____ LAST NAME _____
COMPANY _____
MAIL STOP _____
STREET ADDRESS _____
CITY _____
STATE _____ ZIP _____
COUNTRY _____
TELEPHONE NUMBER (____) _____

CONFERENCE REGISTRATION FEES

	Advance Before 5/14/88	At Conference or After 5/14/88
Member ACM or IEEE	U.S. \$120	\$150
Non-Member	U.S. \$150	\$190
Student (at conference only)	—	\$75

*ACM Member _____

*IEEE Membership _____

This fee includes Monday evening cocktail party, Tuesday, famous DAC party, daily coffee service, one copy of the Conference Proceedings, and free busing.

*This is for individual memberships only. This does not apply to the company memberships. If this individual number is not included, non-member fees will be charged

NEW PRODUCT REVIEWS

Editor: Richard Eckhouse, MOCO, Inc., PO Box A, 91 Surfside Rd., Scituate, MA 02055; Comppmail+, r.eckhouse

Alternative input devices

Richard Eckhouse, New Product Reviews editor

While most of us use the keyboard as a standard input device, we have many other alternatives, the most common being mice, light pens, trackballs, touch screens, and tablets. More esoteric (or eccentric) devices are also available.

Of all the alternative devices, the mouse is probably the most popular. Low in cost and easy to use, the mouse requires relatively little desk space and fits naturally with screen-oriented applications. To use a mouse, however, requires the ability to coordinate what the eye sees and the hand does.

Try switching to a trackball after mastering a mouse. Some operations seem quite natural, although you may notice a tendency to want to move the whole device and not just roll the trackball.

Some maneuvers can be really clumsy with the trackball, like click and drag operations. However, the better trackballs will usually include a cursor drag function that does not require holding down a button while moving the trackball. Since this function is implemented by pressing a combination of buttons, most trackballs have at least three buttons.

The trackball has another advantage: it doesn't require the movement space that a mouse does. And, if the trackball emulates one of the more popular mice, it means that the trackball can directly and transparently replace the mouse. You could plug the trackball into a bus mouse interface card and use the standard mouse software. Alternatively, if you plugged it into a serial port, you can use the existing mouse software because the trackball emulates the mouse. However, trackballs cost more than a mouse, typically because of the lesser volume sold and the smaller number of vendors.

Light pens and joysticks also have many of the attributes described, but for many reasons have not caught on. Probably the primary reason is that both devices require an input port different from the serial port that mice and trackballs use. Of course, software is another important consideration and,

since most applications include only mouse or tablet drivers, the manufacturers of these alternative input devices do not enjoy the widespread support that mice do.

While touch screens are most noted for their appearance on information kiosks and in high-end systems (military and commercial), they generally cost two to ten times more than a mouse and yet have considerably less resolution. A typical 12- to 19-inch screen has 256 × 256 addressable points (about 25 points per inch), whereas a mouse has a minimum of 200 points per inch. Also, a touch screen has no equivalent to the buttons found on mice, trackballs, and tablet pointers.

While generally priced several times higher than a mouse, tablets are becoming real contenders in the PC marketplace. Like the other devices mentioned, much of the popularity of this particular input device seems to depend on several things: ease of use, price, and compatibility with an existing application.

Tablets best suit applications like CAD/CAM, desktop publishing, graphic arts, and data entry by means of menu selection. Tablets work best with these applications for two primary reasons. First, in these applications the user must choose among a large number of alternatives. Second, the alternatives are arranged in a fixed format better suited to a tablet's absolute positioning. For example, in the CAD environment this means descending down menus of items until you choose a specific option. Then, without stopping, you must immediately execute an option and perform specific tasks such as drawing an object; dimensioning, transforming, or erasing an object; or selecting among the components that make up an object. The catch with a tablet is that it takes up a lot of desk space.

A typical tablet consists of a flat desktop drawing surface ranging in size from 12 × 12 inches to 16 × 16 inches. Not all of the surface is active; the smaller work space occupies from 8 × 8 to 12 × 12 inches. The technology used

to implement the tablet is usually an electric or magnetic field activated when a pen or puck (I will refer to either as a pointer) comes within a certain proximity of the tablet's active area. For most tablets, this translates to within half an inch above the surface.

The tablet pointer usually comes equipped with one or several buttons. As you move the pointer, a stream of x-y coordinates is sent back to the computer via the serial port. The coordinates can be either relative or absolute. To integrate a tablet into a specific application, a software driver must be provided (just like for a mouse).

Two features are quite valuable to have with a tablet: mouse compatibility (such as using the pointer as a mouse) and programmability of the device to allow instant changes as the user moves from one application to another.

In this review we look at a new trackball and two tablets. Each of these devices exhibits the features we've described while adding a few extra touches that make them unusual in their own right. Choosing which one is right for you will depend on the specific application and your preferences.

MicroSpeed FastTrap

When it comes to trackballs, the MicroSpeed FastTrap is in a class of its own. First of all, it's the only device to include a z-axis trackwheel in addition to the normal x-y axes trackball. Second, it's small (about 4 inches wide by 7.5 inches long and 2-2.5 inches high), lightweight, and appears to have been designed to reside to the right (or left) of the normal PC keyboard. It's made of durable plastic shaped to provide a comfortable feel during normal use. And it's a three-button device that offers a number of interesting and useful options.

Because it's compatible with the Microsoft serial mouse, you can unplug that device, plug in FastTrap, and start using it immediately. Power is supplied



The MicroSpeed FastTrap features three-axes capability and claims a larger tracking surface than other trackball devices, with twice the pointing resolution.

through the serial port. Everything works normally except, of course, that you don't need a large working area to use it. In addition, the folks at MicroSpeed solved the click-and-drag problem, as well as the double-click problem, so that using their device is actually simpler than using a mouse. They did this by using the middle button as a drag on/off switch. If you press and hold either the right or left button, and then the middle button, you stay in drag mode. Press any button again and drag turns off. In addition, if you press and hold both left and right buttons, and then press the middle button, pressing the middle button from then on emulates pressing both left and right buttons simultaneously.

In actual use, the device is fun to use and much more convenient than a mouse. For those of us who need some time to get accustomed to a trackball in mouse mode, the z-axis trackwheel can be used to vary the trackball gain. As MicroSpeed points out, "Power users will probably like the more responsive fast gain setting while new users may initially be more comfortable learning to use the pointer on the slow gain setting." To change from one mode to another, you simply rotate the trackwheel up or down one revolution. The gain settings are slow, normal, and fast, and correspond to 50, 100, and 200 pulses per inch.

So much for making this device every bit as easy to use as a mouse. In actual operation, you will generally find the trackball convenient when you need to accurately position the cursor and then take some action. A mouse typically

tracks off the work space, requires you to lift and reposition the device, and then moves just as you release it or attempt to press a button. A trackball, on the other hand, allows you to pan over your application by rolling the trackball in the palm of your hand, then use your fingertips to precisely position it where you want it. Because the device sits in a stable position on your desk, it doesn't move when you lift your hand or try to press a button.

Besides supporting the Microsoft mouse standard with its set of compatible function calls, the FastTrap trackball includes its own set of extended function calls. These function calls make it possible to take full advantage of this three-axes pointing device. However, the newness of the device means that there are few, if any, applications that use this FastTrap feature. In their well-done manual, the MicroSpeed designers devote a full chapter to future applications in such fields as desktop publishing, word processing, CAD/CAE, spreadsheets, databases, and games. A number of programs come with the system, including both MAP.SYS and MAP.COM, the MicroSpeed device drivers. While you don't need to use these drivers for normal mouse emulation, you will need them when using KeyMap, a keyboard emulator program.

KeyMap is a TSR program that allows you to make full use of FastTrap with applications that do not currently support a pointing device. A number of templates already included with KeyMap translate FastTrap operations into equivalent keyboard operations. For

example, using the predefined DOS template, you can move the trackball in the x-y plane and get the same result as pressing the arrow keys. Moving the trackwheel acts the same as pressing the page-up or page-down keys. Pressing one of the three buttons is equivalent to hitting the Return, Esc, or F3 keys, respectively. In combination with the shift, Alt, and Control keyboard keys, the buttons can take on up to 12 different keycodes.

I tried using KeyMap with the Norton Editor, DOS, PCTools, and Leading Edge Word Processor and really liked the results. Except for Leading Edge, which takes control of the keyboard interrupt and therefore did not work, each of the other applications was much easier to use under KeyMap. Rapid scanning of directory lists, menu selection, and keyboard responses were easy to accomplish using only the trackball. Part of the reason for this is that you can place your hand on the FastTrap device without really looking at it, and then, with eyes directed at the screen, make your choice.

Up to 32 templates can be stored with each activation of KeyMap. MicroSpeed includes several for such programs as Lotus 1-2-3, WordPerfect, WordStar, dBase, and Turbo Pascal. You can customize each of these templates as well as create your own using the KeyMap edit function. You can also disable KeyMap for applications requiring mouse emulation.

MicroSpeed just recently began shipping its own driver for AutoCAD. With this new driver, you can select menu items directly, using the trackwheel in combination with the middle button. The result is that you no longer need to move the cursor out of the active drawing area and into the menu area to select a menu item.

I'm sold on this trackball and find that I prefer it to my mouse. It's completely compatible with DESQview and the applications running under that window manager. It's also compatible with all the other applications I use, including MS-Word and CADKEY. At \$149, it's slightly more expensive than a mouse, but it surely packs in more features.

Reader Service 20

Summagraphics Bit Pad Plus

Summagraphics is probably the best known supplier of digitizing tablets. The company has been a leader in developing low-priced digitizers, and



Summagraphics reportedly designed its Bit Pad Plus to be a low-cost, full-sized graphics tablet with precise drawing and tracing capabilities.

the SummaSketch is often taken as the standard against which to measure other tablets. It is also generally the first tablet supported by new drawing or CAD/CAE packages. In a very true sense, Summagraphics has become its own competition.

In designing a new tablet, Summagraphics wanted it to be full-sized with precise drawing and tracing capabilities. Lower cost was an important goal as well. There is no question that they met all of these goals in the Bit Pad Plus. Equally important was ease of use, which translates into a "plug and play" integration with existing applications.

When it comes to installing and using a tablet, the new Summagraphics Bit Pad Plus is a snap. All you have to do is plug either the stylus or four-button puck into the tablet, connect both the tablet and the power supply to the adapter plug using the cables supplied, and plug the adapter into your PC using its 25-pin serial port. All of this is explained in great detail in the illustrated user's manual. However, it's darned near impossible to make a mistake hooking things up, so looking at the manual would be the last thing most of us would do.

When you turn on the power to the tablet, it immediately calibrates itself. During calibration, the power light blinks to indicate what's going on. From then on, the light serves as a proximity indicator, steady on when the pointer is in range and blinking whenever the pointer is out of proximity. This useful feature is complemented by the beeping of the tablet whenever a

button on the puck or the tip of the stylus is pressed. I think every tablet should have these features and I hope other manufacturers take note of it.

Using the Bit Pad Plus, once physically connected, is simply a matter of selecting it from the list of devices supported by the applications software for which it will be used. No software is supplied with the tablet, so, if you want to use it with an application that doesn't support it, you will have to write your own driver. While Appendix A of the user's guide does specify the baud rate and data format, you will need to order some additional manuals from Summagraphics if you choose to write your own driver.

The resolution of the Bit Pad Plus is up to 254 lines per inch and the data rate is fixed at 9600 baud. When used with the CAD system from CADKEY, the tablet worked flawlessly. Menu selection and cursor movement were smooth and accurate. The button click was loud enough to be heard above the noise of the computer and distinctive from the keyboard click. The nonmarking stylus was easy to hold and glided smoothly across the 12-by-12-inch active drawing area. Overall size is approximately 16 inches wide by 17 inches tall, with a slight slope front to back. The tablet is light enough to hold on your lap, and is completely enclosed in a high-impact molded plastic case.

Summagraphics uses Charge Ratio as its technology in the new Bit Pad Plus. The company claims that this technology offers the lowest cost and highest reliability in a digitizing tablet. Cer-

tainly, the single-board design and fewer parts should make this a low-maintenance, high-reliability device. At its current price of \$495, which includes both the stylus and four-button puck, the tablet is certainly a good value. I would expect this price to drop in the near future, in part because of the competitiveness of this marketplace and the economies that come as production ramps up.

In summary, the Bit Pad Plus is a well-made, no-frills tablet. If you need a low-cost device, supported by nearly every application, that you can take out of the box and use immediately, then this is the tablet to buy.

Reader Service 21

Kurta IS/ONE

While at first glance the Kurta IS/ONE may look like an ordinary tablet, it is anything but. A close look reveals a series of soft switches across the top of this device, while along the back panel are three dip switches, offering a multitude of tablet settings. There is no external power supply. Also, the eight-degree slope of the tablet is noticeably steeper than found on other similar tablets.

Common to the IS/ONE and other tablets is the molded plastic case, choice of stylus or four-button puck, and 12-inch-square drawing surface. The differences start, however, with its three-button, cord-free stylus, as well as a cord-free model of the four-button puck.

The most significant features that distinguish this tablet from the others are its programmability and extensive use of soft keys. To understand this, we need to first examine the dip switches in the back of the unit that make this a multiple personality tablet.

The first group of switches determines the mode and hence how the tablet sends data to the computer. Normally the tablet is in auto mode, which means that data is continuously output when the pointer is within proximity of the tablet surface. Five other modes are also possible.

The next group of switches determines the menu located at the top of the tablet. The last group within the first dip switch selects the baud rate, ranging from 300 baud to 19.2 kilobaud, with auto speed recognition possible. In the second dip switch, the data format (including encoding, number of data bits, and parity) and rate (conversions per second) are specified. The last dip

switch sets the resolution (200 to 1000 points per inch in English or metric units), CTS signal recognition, and emulation mode (for prior Kurta tablets or even tablets bigger or smaller than the one currently being used).

How these dip switches are set determines the default setting for the tablet on power up. If you change from one application to another and need to use a different set of settings, it is not necessary to turn off the unit and start all over. First, if you want one of the default settings stored within the 16-bit CMOS processor built into the tablet, you merely have to touch one of the soft keys labelled P1 to P5 to instantly make the change. For example, one of these soft keys makes the tablet look like a Microsoft mouse; another makes it look like a Summagraphics MM series tablet. If the preprogrammed soft keys don't offer what you want, you can download any one of them specifying every possible option.

Even more custom tailoring is possible. In addition to the configuration keys, there are pen function keys that work with the dual action stylus. The "soft" setting sets up the pointer for drawing, while the "snap" aids menu selection. If you desire, you can even enable both.

Finally, we come to the function keys. These 13 soft keys can be programmed to transmit any series of keystrokes you want. Thus, you can tailor the keys for a particular application. In the case of CADKEY, I used this capability to load the soft function keys with

CADKEY's immediate commands (Alt-A for auto scale or Ctrl-X for cursor snap). By doing this, I could avoid going to the keyboard for frequently executed commands. As you can imagine, this really made me more productive, since I needed less access to the keyboard.

With all this functionality, you'd think IS/ONE would be a very expensive tablet. It's not. Depending on tablet size, configuration, and options, prices range from \$565 to \$1145. In my case, the price for the tablet was \$495, plus \$50 for the IS/ONE AT kit, \$100 for the four-button puck, and \$150 for the cord-free pen.

Besides this hardware, you get a diskette containing a series of drivers and a configuration program. The configuration software allows you to define whether the tablet is used in relative or absolute mode, the equivalent keystrokes for the function keys, the active area, and the communications port it's connected to. Gem and Ventura installation instructions are also included so that you can set the tablet to work with these systems. Other software includes an interface to Microsoft Windows, a software development kit, and templates for Aldus PageMaker, AutoCAD, and VersaCAD.

One thing about the Kurta tablet could use improvement. The manuals read like engineering specifications rather than user's guides. While all the necessary information is packed into the manuals supplied with the tablet, they are very hard to read. It must have

taken me at least three readings in some cases to make sure I understood exactly what to do or what features and functions were available for use.

IS/ONE is an alternative input device for the tablet aficionado. For a little bit more money than a standard tablet costs, you get programmability and versatility, and a device that can work in practically any environment requiring either a mouse or a tablet. This versatility alone can justify the additional cost.

Reader Service 22

Recommendations

All of the devices discussed here are excellent alternatives to the standard keyboard. People unaccustomed to the hand-eye coordination that a trackball or tablet requires will need some "training" time to master the art of using these new devices. Confirmed mouse users should be able to make the adjustment quickly.

I chose to look at three devices ranging in price from \$149 to \$1145. Each offers price-performance advantages that depend on the user's needs and the application environment. Thus, it's difficult to recommend one over the other. Suffice it to say that no matter which one you choose, you will enjoy these devices and they will make your interaction with your application more efficient.

Entry-level networking system

Dick Eckhouse, New Product Reviews editor

A promising new product from Trans-M Corp. of Medfield, Mass., offers low-cost local area network connections between IBM PCs and compatibles. Using CSMA/CD and a proprietary protocol, the Net-127 system links up to 127 systems at 250 kilobits in an easy-to-use and easy-to-install LAN. The hardware consists of a short board and 25 feet of telephone cable with RJ-11 jacks on each end. The software comes on a floppy and includes an installation procedure.

I installed the boards in several Compaq computers, following the installation procedure to assign a station ID and number of remote drives. I fixed up the CONFIG.SYS and AUTO-EXEC.BAT files to include this new device. I ran the NCONFIG program to

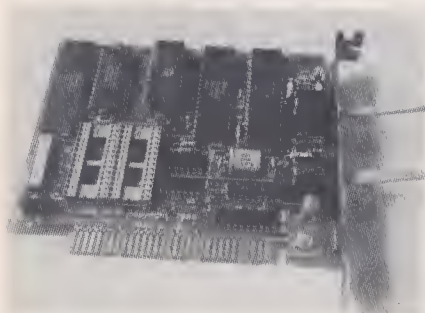
assign device letters, access level, and drive assignments as required. Finally, I assigned the printers and the single COM port, again indicating accessibility. I did all of this with only a cursory glance at the manuals, attesting to the ease-of-use of this product. The manuals, while brief, do cover all of the installation and use in detail, and include a lot of troubleshooting information for those who may need it.

I tried out the network in a couple of typical LAN applications. In the first case, I used one node, with several hard disks, as a file server to another node. Things worked well, although the time it took to remotely load a large EXE file was noticeably longer than if the file had been loaded locally. Obviously, the speed of the network was the limiting

factor. What this case did illustrate was the feasibility of the approach and that a low cost system can be very functional.

My second test was more typical: I used the net as a means of remotely accessing data. Here, performance was excellent, and the application worked exactly as it would have if the data had been stored locally.

In my last test, I used the net to access a remote printer. Because the speed of the network was faster than the bandwidth of my dot matrix printer, there was virtually no penalty for the remote access. The bottom line is that everything worked well, completely transparent to the application and to me, the user. Of course, I did have to give up 32 kilobytes of RAM on each



The Net-127 from Trans-M connects IBM PCs and compatibles, using CSMA/CD and a proprietary protocol. Net-127 consists of software and an adapter card, shown here.

machine for the memory-resident Net-127 software, but this seemed a rather modest sacrifice.

I did have some minor problems that may have been unique to my configuration. On my XT Compaq Deskpro, I couldn't find an interrupt and port address that wasn't taken up by the

several additional boards, such as a bus mouse and second hard card I had previously installed. On my Compaq 286 Deskpro, I ran into a conflict with the Intel Above Board. In both cases, Trans-M technical support was most helpful, thanked me for the information I provided, and gave me several options to work around the problems. One solution would have been to modify their boards, something I chose not to do. The other was to simply pull out the offending boards during my tests; this was a simple way to get going quickly and with little hassle (and was my obvious choice). This should not be a problem in the future, I was informed, because the next release of the system will solve both of these problems.

I really would have liked to have kept this system installed. In a small shop like ours, linking machines sure beats the current method of moving floppies from machine to machine, and avoids the double density-high density floppy problems when exchanging diskettes between AT and XT class machines.

The savings in time, and the obvious savings in not having printers attached to each machine, would more than justify the price of \$249 per station. However, there are two reasons that I will have to wait. First, the hardware conflicts in the two Deskpros have to be resolved. In all fairness to Trans-M, this is not entirely their doing, but is typical of the "open architecture" lack of standardization in the PC. Second, as the Trans-M board stands, I had to give up either COM1 or COM2 for use by the Net-127 system. Since I often have devices attached to both ports, this presents a problem. As I've already said, Trans-M plans to change things in the next version of its hardware and software, thereby allowing a wider selection of interrupt and port addresses.

I hope to be able to give an updated report in a future column. This is an exciting and low-cost product that should and could be a part of every PC system where there is an obvious need to share devices.

Reader Service 23

Another visit with the Living C interpreter

Daniel McAuliffe, McDonnell Douglas Astronautics Co.

I first reviewed Living C Personal for the March 1986 issue of *IEEE Software*. Since that time, the product has been renamed Living C Plus, a number of changes have been incorporated, and new features have been added.

The most visible change is the addition of a new menu and windowing system. Pull-down menus let you access program functions, with user-definable shortcut keys available for the more commonly used options. All text and messages in Living C are viewed through windows. You can move and resize any window and also control the window color scheme. The new menu and windowing system resembles that found in other products in the micro-computer marketplace. It is easy to use and a great improvement on the previous interface.

The program editor has received a major overhaul. You can now edit up to 12 files of any size at the same time. Each file is assigned to a separate user-configured window. Data is moved between files with a cut and paste buffer. The editor also includes a powerful undo feature. You can erase as many as 35 previous actions. Another nice feature is the auto-save function. You can specify an elapsed time, in minutes, after which the editor will automatically save to disk any files

that have been modified. This includes any file in the list of files you may currently be editing.

The number of library functions provided with the new version of Living C has been increased from approximately 50 to more than 140. All functions conform to the proposed ANSI standard.

The help facilities have been expanded from the previous version. They are excellent in every respect. Information on program compilation errors often includes examples of the types of usage that may have generated the error—a great improvement over the cryptic messages generated by many compilers.

The documentation supplied with Living C is much improved over the previous version, although it still contains an annoying number of small errors, such as discrepancies between figures and text. It could easily tolerate another round of proofreading.

This new version still requires an IBM or 100-percent-compatible PC with DOS 2.0 or higher to run. However, the amount of memory needed is now 512 kilobytes.

The most outstanding features of Living C are still the debugging and tracing facilities. The animation mode is valuable not only for finding obscure

program errors, but for insight into the way C expressions are evaluated.

In addition to animation mode, Living C includes support for breakpoints, single stepping through expressions using a single keystroke, and monitoring and changing the values of program variables. These functions were present in the previous version, but ease of use has improved considerably with the new menu and window system.

About the time I received the current version of Living C for review, I also received a copy of the Microsoft Quick C compiler. This made it very difficult to evaluate Living C without some comparison with Quick C. Although the Living C trace and debug features hold up well against the Quick C debugging features, it has a difficult time competing with Quick C when it comes to additional features and price. Quick C offers a superior development environment, program make facilities, extensive graphics functions, and an upgrade path to the more powerful Microsoft V5.0 compiler. Quick C also sells for \$100 less than Living C.

The new version of Living C is much improved over the previous version and offers a number of interesting features. However, at \$200 it may be too little for the additional cost.

Reader Service 24

HOW TO JOIN

Membership dues and publication subscriptions are annualized to December 31
Pay the full-year fee if application is mailed September 1–February 29.
Pay the half-year fee if application is mailed March 1–August 31.

1 COMPUTER SOCIETY ONLY (affiliate membership).

You are eligible if you are seriously interested in any aspect of the computer field or if you are a member of one of the affiliate societies listed below.

(Check all applicable societies.)

Affiliate Societies

- | | Half-Year
Mar 1-Aug 31 | Full-Year
Sept 1-Feb 29 |
|--|----------------------------------|----------------------------------|
| ☐ American Institute of Aeronautics and Astronautics (AIAA) | ☐ \$19.50 | ☐ \$39.00 |
| ☐ American Institute of Physics (AIP) | | |
| ☐ American Mathematical Society (AMS) | | |
| ☐ American Society of Mechanical Engineers | | |
| ☐ Association for Computing Machinery (ACM) | | |
| ☐ Australian Computer Society | | |
| ☐ British Computer Society | | |
| ☐ Data Processing Management Association (DPMA) | | |
| ☐ Information Processing Society of Japan (IPSJ) | | |
| ☐ Institute of Electronic, Information and Communication Engineers (IEICE) of Japan | | |
| ☐ Institution of Electrical Engineers (IEE-UK) | | |
| ☐ Instrument Society of America (ISA) | | |
| ☐ Mathematical Association of America | | |
| ☐ National Association of Accountants | | |
| ☐ Operations Research Society of America (ORSA) | | |
| ☐ Society of Aircraft Materials and Processes (SAMPE) | | |
| ☐ Society of Automotive Engineers (SAE) | | |
| ☐ Society for Industrial and Applied Mathematics (SIAM) | | |
| ☐ Society for Computer Simulation (SCS) | | |
| ☐ Society for Photo-Optical Instrumentation Engineers (SPIE) | | |
| ☐ The Computer Society of the Republic of China (Taiwan) | | |

3 COMPUTER SOCIETY MEMBERSHIP FOR IEEE MEMBERS.

If you are presently an IEEE member, you may join the Computer Society for a nominal amount. (Complete only shaded area of application.)

IEEE Member Number _____

	Half-Year Mar 1-Aug 31	Full-Year Sept 1-Feb 29
☐ \$ 7.50	☐ \$ 7.50	☐ \$15.00

OPTIONAL PUBLICATIONS

4 Whichever membership option you chose, you are now eligible to subscribe at low member rates to these periodicals.

	(issues per year)	Half-Year Mar 1-Aug 31	Full-Year Sept 1-Feb 29
IEEE Computer Graphics & Applications (3061)...	6	☐ \$ 9.50	☐ \$19.00
IEEE Design and Test (3111)	6	☐ \$10.00	☐ \$20.00
IEEE Expert (3151)	4	☐ \$ 6.00	☐ \$12.00
IEEE Micro (3071)	6	☐ \$ 9.00	☐ \$18.00
IEEE Software (3121)	6	☐ \$ 8.50	☐ \$17.00
Transactions on Computers (1161)	12	☐ \$ 9.00	☐ \$18.00
Transactions on Pattern Analysis and Machine Intelligence (1351)	6	☐ \$ 7.50	☐ \$15.00
Transactions on Software Engineering (1171)	12	☐ \$ 9.00	☐ \$18.00

TOTAL

(remittance in U.S. dollars only)

☐ Check or Money Order
(payable to the IEEE)

☐ Visa

☐ Master Card

☐ American Express

Credit Card Number _____

Expiration Date _____

STUDENTS! ▶

You receive special rates and benefits. Call or send for a student application.

2 COMPUTER SOCIETY AND IEEE.*

In addition to your Computer Society benefits, you'll receive many IEEE privileges and benefits. You are eligible if your technical interests are in computer science and engineering, the electrical and electronics fields, or related fields. Your entry membership grade will be determined by your level of participation, contributions, education and/or experience in those fields.

Residence

	Half-Year Mar 1-Aug 31	Full-Year Sept 1-Feb 29
United States	☐ \$41.00	☐ \$82.00
Canada	☐ \$38.50	☐ \$77.00
Europe, Africa, Middle East	☐ \$39.00	☐ \$78.00
Latin America	☐ \$33.50	☐ \$67.00
Asia, Pacific	☐ \$34.50	☐ \$69.00

*ACM members who join both IEEE and the Computer Society are entitled to a deduction of \$5 off the full-year rate; \$2.50 off the half-year rate.

I hereby make application for Computer Society membership and, if elected, will be governed by IEEE's and the Society's Constitutions, Bylaws, and Statements of Policies and Procedures.

MAILING ADDRESS

Full signature _____

Date _____

First Name _____ Middle initial(s) _____ Last name _____
Street address _____ City _____ State/Country _____ Postal Code _____

Were you ever a member of the IEEE? ☐ Yes ☐ No ☐ Grade _____
If yes, please furnish _____ Membership No. _____ Date of birth _____ ☐ Male ☐ Female

EDUCATION (highest level completed)

Course _____ Degrees received _____ Date _____

Name of educational institution (Please print full name) _____

REFERENCE (an IEEE member; if unknown, a managerial person who knows you professionally)

OCCUPATION

Title or Position _____

Name (print in full) _____

IEEE Member No. (if applicable) _____

Firm Name _____

Position _____

Firm address _____

Street Address _____

City _____ State/Country _____ Postal Code _____

City _____ State/Country _____ Postal Code _____

Residents of Europe mail to:

Computer Society
of the IEEE
13, Avenue de l'Aquila
B-1200 Brussels, Belgium
Telephone: 32-2-770-21-98

Payment to the Brussels office may be made by cheques in Belgian francs, British pounds sterling, German marks, Swiss francs, or US dollars, or by American Express, Eurocard, MasterCard, or Visa credit cards.

All others mail to:

Computer Society
of the IEEE
10662 Los Vaqueros Circle
Los Alamitos, CA 90720-2578
Telephone: (714) 821-8380

Allow six to eight weeks
for delivery of publications.

NEW PRODUCTS

Delta FPP card helps PC-AT handle neural nets

The Technology Research Group of Science Applications International Corp. has announced the Delta Floating Point Processor accelerator card for the IBM PC-AT. According to SAIC, the card, when combined with window software, allows users of the company's Artificial Neural Systems Simulation Software (ANSim) to build neural networks on the PC-AT.

The Delta FPP reputedly achieves peak speeds of 22 Mflops. It comes with

12M bytes of memory and supports 32- and 64-bit floating-point and integers. The card costs \$14,950.

Software for the Delta FPP includes 13 user-configurable neural models in ANS simulation, provided in the company's ANSkit. The 13 paradigms included in ANSkit are written in C under Microsoft Windows. (See New Products in the January issue of *Computer*, p. 93, for a description of ANSkit.) ANSkit costs \$995.

Also available for the board are a C compiler, an assembler, and an actor-based language called ANSpec.

Technology Transfer will sponsor "Neural Networks for Artificial Intelligence" to be held in Washington, D.C., March 21-23. Geoffrey Hinton of the University of Toronto will lecture during the three-day course, which costs \$895. Participants will manipulate the neural models provided in SAIC's ANSim software. Telephone (213) 394-8305 for more details.

Reader Service 30

Anza-Plus turns a PC-AT into a neurocomputer

Hecht-Nielsen Neurocomputers offers the Anza-Plus Neurocomputing Coprocessor System, which reputedly transforms an IBM PC-AT or 80386-compatible into a neurocomputer capable of real-time processing. The board targets both applications development and delivery.

The company claims that the Anza-Plus can implement neural networks with up to 2.5 million processing elements and interconnections, and can update the network at a peak rate of 10 million interconnects per second or at a sustained rate of 6 million IPS in the feed-forward mode, 1.5 million IPS in learning mode (back propagation).

The board comes with Release 2.0 of the company's Neurosoft software, which reportedly enables users to treat neural networks as subroutines within C programs. The software includes the User Interface Subroutine Library and five neural net packages. All neurocomputing functions are executed within the on-board memory of the Anza-Plus, leaving the host free for I/O functions and preprocessing.

The coprocessor board comes in two versions, one with 10M bytes of memory and one with 2M bytes of memory. The 2M-byte model is limited to implementing neural networks with a combined total of 500,000 processing elements and interconnects.

The 10M-byte model costs \$14,900. It also comes bundled in a Zenith Model 248 PC for \$19,900 or in a Zenith 386-80 PC for \$24,900. The 2M-byte

model costs \$3900 less in all of these configurations.

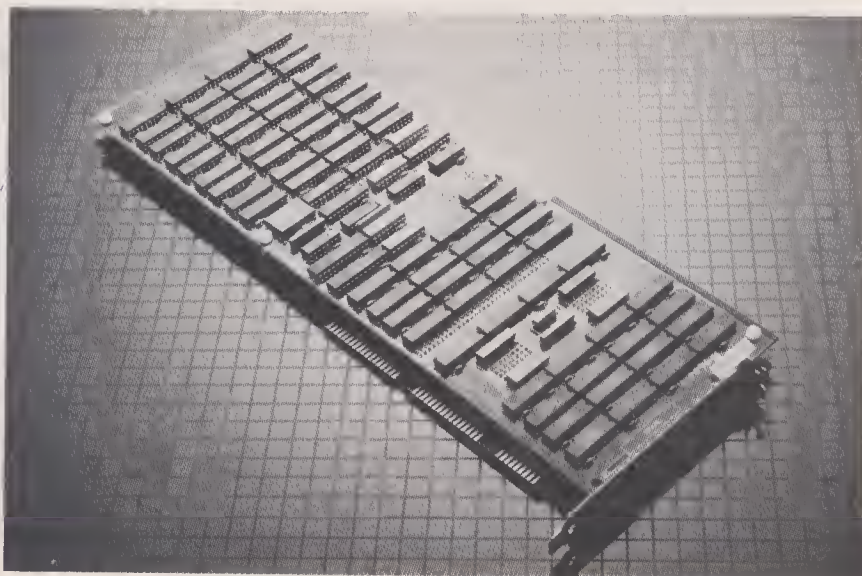
The Zenith 248 host PC comes with a 10-MHz 80286 processor, 40M-byte hard disk, 1M byte of extended memory, and an 80287 coprocessor. The Zenith 386-80 host PC comes with a 16-MHz 80386 processor, 1M byte of RAM, and an 80M-byte hard disk with a 25-ms access time. Both host PCs include a 1.2M-byte floppy and an EGA color adapter with 13-inch color monitor.

The Anza-Plus is based on the Weitek Accel family of processors, which

incorporates reduced instruction set computer technology. Processor features include a 32-bit floating-point processor capable of 20 million floating-point operations per second. The board has a peak power dissipation of 15 watts.

Hecht-Nielsen Neurocomputers has reduced the price of the original Anza coprocessor to \$6900, including Neurosoft software.

Anza-Plus: Reader Service 31
Anza: Reader Service 32



The Anza-Plus from Hecht-Nielsen Neurocomputers is a single-board coprocessor for implementing neural networks on an IBM PC-AT or 386 compatible.

Nestor Development System emulates human learning

Nestor, Inc. has announced the Nestor Development System, a neural-network, artificial-intelligence-based product which the company says emulates aspects of the human thinking process and learns automatically from its own experience. The system reportedly eliminates the task of knowledge engineering and rule programming on which AI expert systems are based.

The company claims that NDS simplifies the creation of neural architectures, and that its performance in learning and recognizing new patterns is independent of the number of patterns previously learned. The software features the proprietary Nestor Learning System (NLS) multimodule neural network architecture, recognition packages for a variety of pattern recognition applications, diagnostics for system configuration and development of new coding methods, examples of complete recognition packages, and documentation on how to apply the NLS recognition technology.

Nestor describes the software's components as tools to enable you to train new NLS memories, tabulate recognition performance statistics, determine which training patterns are leading to erroneous results, and evaluate the need for revising your feature-extraction methods. Using NDS reportedly involves seven steps:

- (1) Selecting the recognition problem and collecting sample data
- (2) Developing feature-extraction and encoding methods
- (3) Defining the NLS system architecture
- (4) Training and testing the NLS recognizer with sample data
- (5) Evaluating recognition results

(6) Revising the feature-extraction methods and system architecture to improve recognition results

(7) Merging the recognition package with an application program

NDS code is written in the C language. Portions of the training program and code explained by the tutorial are supplied as source code.

The system features 15 million connections, 150,000 processing elements, 500,000 connections per second, and dynamically allocated connectivity, according to the company.

The NLS-callable routines require 50K bytes of memory. NLS memory requirements are application dependent.

NDS runs on Sun or Apollo workstations or the IBM PC-AT. It requires a 1.2M-byte disk drive, 512K bytes of memory, MS-DOS 3.0 or higher, Microsoft C version 4.0 or higher, and a 10M-byte hard drive.

The total cost of NDS on the Sun and Apollo workstations is \$25,000. This breaks down into \$9000 for a two-week training period and \$16,000 for the first copy of the software. Additional copies for inclusion in applications cost \$10,000 or less, depending on quantity. The training concentrates on background information, followed by development of the customer's specific application. The IBM PC-AT version costs \$8,500 for the software, for a total of \$17,500 with training. Additional copies cost \$5000 or less, depending on quantity.

NDS was developed by Leon Cooper and Charles Elbaum, with further development proceeding in cooperation with Douglas Reilly and Christopher Scofield of Nestor.

Reader Service 33

Concurrent 3280SP enters superminicomputer market

Concurrent Computer Corp. offers the Model 3280SP uniprocessor superminicomputer. The system comes in a 71-inch cabinet with up to 1G byte of disk storage, up to 32M bytes of memory, and up to 20M-bytes per second of data throughput, according to the company.

A base system features disk and magnetic tape subsystems with a 4M-byte memory capacity for \$199,500. The company claims a processing performance for the Model 3280SP of 6.4 million single-precision Whetstone instructions per second.

The 3280SP operates on Concurrent's proprietary operating system, OS/32, and the Xelos operating system, a port of Unix System V.

Other features include a 32-bit processor with 16K-byte cache memory, 64-bit floating-point processor, and 8K-bit words of writable control store; a 64M-byte-per-second memory bus; an HPD-368F disk; and a 10M-byte-per-second direct memory interface, expandable to two interfaces with an aggregate of 20M bytes per second.

Reader Service 34

Prime offers MXCL 5

Prime Computer has introduced the MXCL 5 departmental supercomputer for numerically intensive computing. The new minisupercomputer comes from a partnership with Cydrome Inc., which designed the system's Directed Dataflow architecture.

The MXCL 5 reportedly provides an interactive environment based on AT&T's Unix System V.3 and incorporates a number of standards: IEEE 754 floating-point format, VME bus, Fortran 77, and IEEE 802.3 Ethernet and TCP/IP protocols. It operates as a dedicated floating-point computation engine, or integrated into a variety of networked environments as a computation server for workstations, superminicomputers, or mainframe systems, according to the company.

Features include a 100K, ECL-based, 64-bit numeric processor with a 40-ns clock speed; up to six 32-bit interactive processors with a 16.67-MHz clock speed, for noncomputational functions; a 100M-byte-per-second system bus; an enhanced Fortran 77 compiler; up to 256M bytes of very high bandwidth main memory; up to 64M bytes of support memory; an aggregate 400M-byte-per-second bandwidth for disk access; and I/O performed independently of the numeric processor.

According to Prime, the MXCL 5 was designed with eight functional units working in parallel to balance the computer's processing performance. Each functional unit is specialized to execute a specific type of instruction, such as floating-point addition, memory access, and address arithmetic.

The standard MXCL 5 system includes a numeric processor, an interactive processor, a service processor and personal computer console, an I/O processor, a VME bus, an I/O central cabinet, an I/O expansion cabinet, a disk drive and controller, a tape drive and controller, the MX/IX Unix operating system licensed for an unlimited number of users, a program debugger, a korn shell, and libraries.

The two base configurations available differ only in the amount of memory offered. The entry-level MX1201 model with 8M bytes each of main and support memory and associated peripherals costs \$579,000. The MX1401 with 64M bytes of main memory and 16M bytes of support memory and associated peripherals costs \$774,000.

System installation for the MX1201 and MX1401 costs \$5000 and \$7000, respectively.

Reader Service 35

IBM Controller speeds communications

IBM has announced the 3745 communication controller, featuring what the company calls its newest and densest logic chips. The controller reportedly allows users to channel-attach to a maximum of 16 System/370 computers, plug in line-interface couplers without shutting off the power, and diagnose and repair components without disrupting system operations.

The IBM 3745 comes with the option of one or two internal central control units (Models 210 and 410, respectively). With two units, one can be shut down for upgrading or servicing while the other continues to function. Each CCU can be configured with 4M bytes or 8M bytes of memory. Cache memory provides 16K bytes of memory for each CCU.

The controller incorporates three application-specific integrated circuitry chips from IBM's new family of chips, each chip holding up to 40,000 circuits. It consists of four subsystems: control, communication, maintenance and operator, and power.

The IBM 3745 Model 210 includes one CCU, a 45M-byte hard disk, two low-speed communication scanners, eight line-interface couplers, 4M bytes of memory, Remote Support Facility modem, and line cables.

Contact the company for more information about pricing and availability.

Reader Service 36

Entry-level models join Unisys 1100/90 mainframes

Unisys has added two new models to its 1100/90 family of mainframe computers. The 1100/91 and 1100/92 Model II SV processors reportedly feature object-code compatibility with current 1100 Series and 2200/200 systems.

The single-processor 1100/91 Model II SV uses 256K RAM chips and comes in 8M-byte or 16M-byte memory units. Including operator console and system control software, it costs \$1,429,000.

The dual-processor 1100/92 Model II SV costs \$2,605,000. It features a maximum system memory of 32M bytes.

1100/91: Reader Service 37

1100/92: Reader Service 38



The IBM 3745 communication controller incorporates logic chips containing up to 40,000 circuits per chip. The Model 210 provides one CCU, while the Model 410 provides two CCUs.

Alliant's new 2nd-generation minisupers based on ACE

Alliant Computer Systems has announced the second-generation FX/40 and FX/80 minisupercomputers. The new systems are based on the company's Advanced Computational Element, a general-purpose, 64-bit vector processor. They replace the FX/8.

The top-of-the-line FX/80 features up to eight vector processors, with a peak rating of 188.8 Mflops. A base configuration FX/80 costs \$299,000. It includes two ACEs, two VME-based Interactive Processors (IPs), 32M bytes of main memory expandable to 256M bytes, a 16-line multiplexer, 1.1G bytes of disk storage, and a magnetic tape drive. Up to six ACEs and 10 IPs can be added.

The FX/40 incorporates up to four processors, with a peak rating of 94.4 Mflops. A base configuration FX/40 costs \$149,000. It includes one ACE, one VME-based IP, 32M bytes of main memory expandable to 160M bytes, 1.1G bytes of disk storage, and a cartridge tape drive. Up to three ACEs and five IPs can be added.

All systems include an Ethernet controller, VME I/O chassis, console video terminal, dot matrix printer, and operating system software along with FX/Fortran and the Alliant Scientific Libraries.

The new systems reportedly offer up to twice the performance of Alliant's

earlier FX/8 Series systems, yet are compatible with those machines and the available applications software packages written for them. The new ACEs are plug-compatible replacements for the Computational Element boards in the first-generation FX/8.

Users can add ACEs, priced at \$59,000 each, to increase system performance and capacity without recompiling or relinking, according to the company.

Alliant offers VAX and workstation integration utilities that run on integral and compatible Interactive Processors, which frees the parallel vector processors for complex computations. VAX integration products include DECnet networking, Digital Command Language emulation, EDT editor, and the X Window System. Workstation integration software includes NFS, NCS, and NeWS. Data communications products include TCP/IP support for Ethernet and support for the Hyperchannel. Optional languages include FX/C, FX/Ada, and Pascal.

Alliant has also released previously proprietary specifications on its instruction set for parallel execution. The company said that it wants to facilitate the development of parallel processing applications.

FX/40: Reader Service 39
FX/80: Reader Service 40

NCR adds four models to Tower family, offers new LAN family

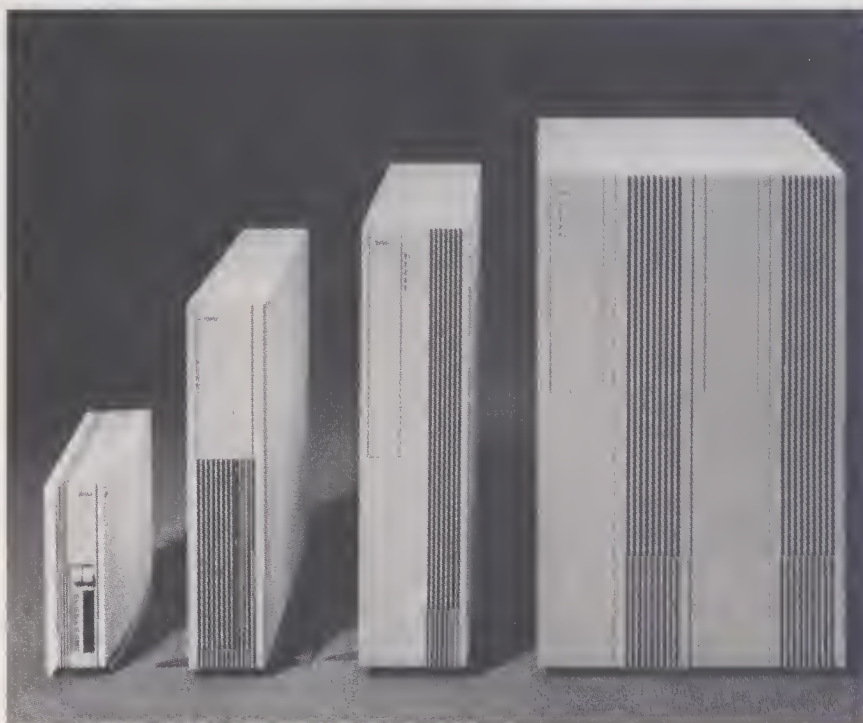
NCR has announced the expansion of the NCR Tower family of Unix-based supermicrocomputers with four new models.

The NCR Tower 32/850 becomes the most powerful member of the family, handling up to 512 users, yet remains compatible with the 32/800. The new system features support for up to six 25-MHz 68020-based application processors, each with 8M bytes of system memory expandable to 16M bytes; a new file processor; up to 64M bytes of system memory available for each AP; 40K-byte cache memory and a 68882 floating-point coprocessor on each AP; and an integrated disk capacity of 1.9G bytes and an external disk capacity of 20G bytes. The 32/850 costs will cost \$106,175, with shipments expected in the third quarter of 1988.

The NCR Tower 32/650 and 32/450 are new midrange models. The 32/450 can support 32 users, while the 32/650 can support 64 users. Both models feature a 25-MHz 68020-based processor with 32K bytes of cache memory. A mass storage controller, storage devices, languages, tools, and communication facilities are available for the new midrange models. Prices will be \$15,565 for the 32/450 and \$24,915 for the 32/650. Volume deliveries are scheduled for June 1988.

The NCR Tower 32/200 is an entry-level system. It supports up to four terminals, networked PCs, and a parallel printer. The 32/200 has a Unix 5.3-based operating system with standard communications and networking protocols, according to the company. The system will come in two configurations: a flex disk-based system and a tape-based system. Available in April, the 32/200 will cost \$5445.

NCR has also announced an NCR Tower local area network family. The NCR Tower LAN Family includes the



The NCR Tower Family of Unix-based supermicrocomputers now includes (left to right) the entry-level 32/200, midrange 32/450 and 32/650, and high-end 32/850.

NCR Tower File Server, NCR Token-Ring Controller, NCR Token-Ring Netbios, and NCR Transmission Control Protocol/Internet Protocol via Ethernet.

The file server reputedly provides a transparent file system between DOS and Unix. It is compatible with Expanded Towner and NCR Token-Ring LANs. Available in the third quarter of 1988, the file server will cost \$1230.

The token-ring controller connects the NCR Tower to a token-ring LAN.

Available in the second quarter, it will cost \$2195.

The token-ring Netbios provides an interface between the NCR Tower File Server and the token-ring LAN. It will be available in the second quarter for \$530.

NCR TCP/IP connects NCR Tower computers to other vendors' computer systems. It will be available in the second quarter.

Tower: Reader Service 41
LAN: Reader Service 42

Adobe offers new production tools

Adobe Systems has announced Adobe Illustrator 88, a Postscript-based product that incorporates tools of the original Adobe Illustrator plus color, a blending tool, resolution-independent pattern fills, and masking. The package also includes a freehand drawing tool.

According to the company, Illustrator 88 lets illustrators see color images on-screen, create color separations for

output to a Postscript typesetter, and print on a color thermal transfer printer. The blending tool permits highlighting, contouring, airbrushing, and shading.

Shipments of Adobe Illustrator 88 are scheduled for May. The software package will cost \$495.

Adobe also offers the Adobe Illustrator Collector's Edition, a package of

pre-built Adobe Illustrator basic graphic shapes. It includes 100 borders and more than 300 dingbats, plus a medium-weight serif and sans serif typeface that can be edited and modified to create logotypes. Available in April, the package will cost \$125.

Illustrator 88: Reader Service 43
Collector's: Reader Service 44

IC Announcements

Company, Model, Function	Comments	R.S. No.
Gould Inc. GC Series Gate arrays	A 1.25-micrometer series of 9 base arrays, part of Gould's ASIC Continuum line. Typical gate delay of 0.5 ns with maximum toggle frequency of 100 MHz for a D-type flip-flop. Offers 1000 to 14,000 usable gates from sea of gates-based arrays of 3000 to 40,000 gates. Packaging includes plastic, ceramic, and TAB with 8 to 280 leads. No prices given.	120
Integrated Device Technology IDT49C460B EDC device	A 32-bit error detection and correction device with maximum error detection times of 25 ns and error correction of 30 ns. A functional replacement for two Am2960s. Cascadable to 64 bits. Generates check bits on a 32-bit data field according to a modified Hamming code. Comes in six 68-pin package options, commercial and military. Cost: \$87.80 (100s) for shrink DIP.	121
Integrated Device Technology IDT721264L30, IDT721265L30 Floating-point chip set	A floating-point chip set: multiplier and arithmetic logic unit. Performs 32-bit operations at 33.4 Mflops and 64-bit operations at 25 Mflops. Pin compatible with the Weitek 1264 and 1265. Features 30-ns clock speeds. Conforms to IEEE STD 754 version 10.0 and has latency time of 120 ns. Packaged in a 144-pin PGA. Cost: \$310 per chip (100s).	122
LSI Logic LDD10000 Direct Drive Series Gate arrays	A family of arrays based on Channel-Free architecture. Six masterslices offered, permitting designs of 8000 to 45,000 usable gates. The company is accepting initial design orders, with customer prototype shipments planned for the second half of 1988. Sample devices will be available in the second quarter.	123
National Semiconductor SCX6B00 Series Gate arrays	Nine gate arrays featuring 0.6-ns typical gate delays and ranging from 400 to 15,000 equivalent two-input NAND gates and 28 pads to 200 pads. Offered in through-hole and surface-mount packages ranging from 8 to 196 pins. Operate over military temperature range. Cost: development charges start at \$12,000.	124
National Semiconductor DP8500 RGP Raster graphics processor	A programmable processor for bitmapped graphics systems. Part of the Advanced Graphics Chip Set. A 20-MHz chip that features a 100-ns bus cycle time on back-to-back vector and block operations. Samples available now; production quantities in the third quarter of 1988. Cost: \$95 (10,000s).	125
NCR 85C20 ECC chip	An error-correcting code chip for optical disk storage devices. Implements a Reed-Solomon error-correcting code. Licensed from Data Systems Technology, which can supply supporting software. Supports data transfer rates of up to 24M bps. Programmable for three to 10 interleaves. Cost: \$23.10 (1000s).	126
Signetics PLC16V8 Series PLD family	Four 20-pin, EPROM-based programmable logic devices designed to replace Series 20 Bipolar PAL ICs. Can be configured to emulate 22 PAL devices in various configurations. A two-level logic element with 10 inputs, 72 programmable AND terms, and 8 output macro cells. Cost: in 100s, \$5-\$9.46 for quarter-power PLC16V8, \$2.67-\$5.08 for half-power PLC16V8.	127
Standard Microsystems LAN-068, LAN-058 Arcnet chip sets	LAN-068 contains the COM9026 LAN Controller, COM90C32 LAN Transceiver, and HYC9068 LAN Driver. LAN-058 includes the COM9026 LANC, COM90C32 LANT, and HYC9058 High Impedance Transceiver. Components are available in plastic. Cost: in 1000s, \$39.60 for LAN-068, \$48.05 for LAN-058.	128
Texas Instruments TMS27PC64, TMS27PC128, TMS27PC256, TMS27PC512 PROMs	Four plastic-packaged one-time programmable PROMs. Available in four densities: 64K (TMS27PC64), 128K (TMS27PC128), 256K (TMS27PC256), and 512K (TMS27PC512). Device speeds range from 150 ns for TMS27PC64 to 200 ns for the others. Cost: in 100s, \$3.16 for TMS27PC64NL, \$3.63 for TMS27PC128NL, \$4.34 for TMS27PC256NL, \$6.16 for TMS27PC512NL.	129
Toshiba America TC511000Z, TC514256Z DRAMs	A family of 1M-bit DRAMs in zip (zig-zag in-line package) organized as 1M-bit $\times$ 1 (TC511000Z) and 256K $\times$ 4 (TC514256Z) with multiplexed address inputs. Available with access times of 84 ns, 100 ns, and 120 ns. Cost: in 1000s, \$19.25 for TC511000Z-12, \$1990 for TC514256-12.	130
Video Seven V7VGA VGA chip	A video graphics array chip with video RAM and VGA hardware compatibility. Supports monochrome and color graphics and text. Operates at dot clock rates up to 65 MHz. Comes with 256K- or 512K-byte VRAM and 256K-byte DRAM. Each memory configuration supports an increasing number of colors with varying resolutions. No prices given.	131

CALL FOR PAPERS



Fifth International Conference on Data Engineering

February 7-9, 1989

Los Angeles, California, USA

sponsored by The Computer Society of the IEEE

COMMITTEE

Steering Committee:

C. V. Ramamoorthy, University of California, Berkeley
P. Bruce Berra, Syracuse University
Gio Wiederhold, Stanford University
Benjamin W. Wah, University of Illinois

General Chairperson:

John Carlis, University of Minnesota

Program Chairperson:

Richard L. Shuey, Rensselaer Polytechnic Institute

Program Co-Chairpersons:

Mohan Ahuja, Ohio State University (Tutorials)
Ophir Frieder, Bell Communications Research
Sushil Jajodia, National Science Foundation
Iris Kameny, Rand Corporation
Raymond Liuzzi, USAF RADC
Gordon T. Ray, NEC America (International Far East)
Gunter Schlegel, University of Hagen (International Europe)
Amit P. Sheth, UNISYS
David Spooner, Rensselaer Polytechnic Institute
Marjorie P. Templeton, UNISYS
Kyu-Young Whang, IBM Yorktown Res. Center

Awards:

David Du, University of Minnesota

Publicity:

James P. Held, University of Minnesota

Treasurer:

Oouglas K. Barry, Control Data Corp.

Local Arrangements:

Homideh Afsarmanesh, Calif. State Univ. D.H.

Industrial Coordinator

Elmer Baldwin, Oracle Corp.

SCOPE

Data Engineering is concerned with the semantics and structuring of data in information system design, development, management, and use; and with computer system and architectural considerations that are relevant to that concern. It encompasses both traditional and emerging issues and applications. The purpose of this conference is to provide a forum for the sharing of practical experiences and research advances from an engineering point of view among those interested in automated data and knowledge management. Our expectation is that this sharing will enable future information systems to be more efficient and effective, and future research to be more relevant and timely.

We are particularly soliciting industrial, business, and government participation. We know it is vital that there be a dialogue between practitioners and researchers. We look forward to reports of information systems experience detailing experiments, evaluation, problems, and opportunities associated with design, implementation, and operation. Such reports will be given special consideration.

TOPICS OF INTEREST INCLUDE

AI and Knowledge Based Systems
Applications and Application Systems
Autonomous Distributed Systems
Communication Systems
Concurrency Control and Data Integrity
Data Access Control and Security

Database Management and Structures
Data Engineering Techniques and Tools
Data Services and Servers
Information System Architecture
Performance Evaluation
User Interfaces

PAPER SUBMISSIONS

Each paper's length should be limited to 8 proceedings pages, which is about 5000 words, or 25 double spaced typed pages. Five copies of completed papers should be mailed before June 15, 1988 to:

Richard L. Shuey, Computer Science Department, Rensselaer Polytechnic Institute, Troy, NY, 12189-3590; [518] 276-8376 or [518] 374-5684; Shuey@MTS@ITSGW.RPI.EDU or shuey@ge-crd.arpa

TUTORIALS

The day preceding the conference will be devoted to introductory tutorials which may provide background for the conference proper. The day following the conference will be devoted to advanced tutorials. Proposals for tutorials on Data Engineering topics are welcome. Send proposals by June 15, 1988 to:

Mohan Ahuja, Department of Computer Science, The Ohio State University, 2036 Neil Avenue, Columbus, OH 43210-1277; [614] 292-6377; ahuja@ohio-state.arpa

COMMITTEE MEMBERS (TENTATIVE)

Peter Apave	Yang-Cheng Hong	Sham Navathe
Adersh K. Anora	David K. Hsiao	Philip M. Neches
Eric M. Aupperle	Chang Hsu	Richard E. Newman-Wolfe
J. L. Beer	Tedee Ichikawa	Dle Dren
Don Batony	H. Ishikawa	Z. Merel Dzsoglyu
Katherine M. Baumgartner	Hermant K. Jain	Christine Perant
G. Belford	Jie-Yong Juang	J. F. Paris
Sharat Shergave	Arthur Kaller	Kritchi Ramamritham
Michael R. Blake	Larry Karschberg	K. V. S. Ramamero
Diane L. Bodwin	K. H. Kim	Petar Rathmann
Richard Breegger	Dan Kogan	Oavid Reiner
Walter A. Burkhard	Walter Kohler	Alain F. Robert
C. Robert Carlson	Robert R. Korfhage	Gruis-Catalin Roman
Nick Carcone	Tosiyasu L. Kunii	Domenico Secce
Peter Chan	Winfried Lamersdorf	Giovanni Marie Sacco
Hong-Tai Chou	James A. Larson	Sharon Selvetar
David Choy	Matt LaSaine	Philip Sheu
Wesley W. Chu	Harbert M. Lee	Edger Sibley
Peter Dadam	Francie Leung	Oavid Stampale
George O. Detlefsen	Phillip M. Lewis	Peter M. Stocker
Earl F. Ecklund, Jr.	Guo-Jia Li	Michael Stonebraker
Ramez ElMasri	Victor Li	Stanley Y. W. Su
David W. Embley	Yao-Nan Lian	Denji Tajima
Michael Evangelist	Lee Lilien	T. C. Ting
Don Feetz	Jane Liu	A. M. Tjoa
James P. Fry	Michael M. T. Liu	Mee Tauchye
Hector Garcia-Molina	Vincent Lum	Yoshioe Udagawa
Georges Gardarin	Stephen F. Lundstrom	Joseph Urban
Robert Garber	Yuen-Wah Ma	Susan O. Urban
Sekti P. Ghosh	Memoru Maekawa	Patrick Veldumez
Mark Goodley	Fred Marynski	Ralph F. Wachter
Georg Gottlob	Gordon McClell	Jack H. Westbrook
Walter Grestidge	John McCarthy	Chao-Chin Yang
Laure Hees	Toshimi Minoura	S. Bing Yeo
Martin Hardwick	J. Eliot Moses	Clemant Yu
Lee Hollar	Tedee Murata	

CONFERENCE TIMETABLE AND INFORMATION

Papers due:	June 15, 1988
Tutorial Proposals Due:	June 15, 1988
Acceptance Letter Sent:	September 15, 1988
Camera Ready Copy Due:	November 1, 1988
Tutorials:	February 6 and 10, 1989
Conference:	February 7-9, 1989

For further information contact the General Chairperson, John Carlis, Computer Science Department, University of Minnesota, 207 Church Street SE, Minneapolis, MN 55455 [612] 625-6092; carlis@umn-cs.arpa@relay.cs.net

AWARDS, STUDENT PAPERS AND SUBSEQUENT PUBLICATION:

Awards will be given to the best paper and to the best student paper (denoted as such when submitted solely by students). The latter will receive the K. S. Fu award honoring one of the early supporters of the conference. Up to three grants of \$500.00 each will be available to help defray travel costs of student authors. Outstanding papers will be considered for publication in the IEEE Computer Society publications: Computer, Expert, Software, and Transactions on Software Engineering, etc. For more information, contact the general chairman.



THE COMPUTER SOCIETY
OF THE IEEE



THE INSTITUTE OF ELECTRICAL
AND ELECTRONICS ENGINEERS, INC.

Microsystem Announcements

Company, Model, Function	Comments	R.S. No.
Analog Devices RT1-980 Array processor	An array processor for Multibus II. Provides numeric processing up to 8 MIPS with 16-bit fixed-point format. Uses the ADSP-2100 DSP microprocessor to provide ALU, MAC, and shift register functions. An 80188 coprocessor manages I/O, DMA, and memory operations. Software development tools and target environment tools available. Cost: \$5995.	135
Applied Data Sciences PCHSD Emulator	Emulates a Gould HSD11 card on an IBM PC-AT or compatible. Features an on-board FIFO, burst transfer rates of 6.6M-bytes per second to or from the high-speed data interface, bidirectional transfers, and software I/O driver routines. Plugs into a dual connector slot in the PC. Cost: \$4675.	136
Columbia University Kermit v. 2.30 Protocol software	An error-correcting protocol for transferring sequential files between computers of all sizes over asynchronous telecommunications lines. Not in the public domain, but not licensed or copy protected. Not available for commercial use. Version 2.30 runs on the IBM PC family, including PS/2 series, and compatibles. Cost: a distribution fee.	137
Flexus International Cobol sp11, v. 1.0 Cobol screen manager	A development tool to support prototyping and user interface management for PC Cobol applications. Allows users to create models without writing Cobol source code, then convert models into Cobol programs. Includes a panel painter, panel file manager, prototyping tool, runtime unit, and source code generator. Cost: \$395.	138
Floating Point Systems MP32 SSP Array processor	A 32-bit array processor with up to four computational processors and a peak speed of 72 Mflops. Available with Digital Equipment, Data General, and Concurrent interfaces, including Qbus, Unibus, VAXBI bus, Burst Multiplex Channel, and BSELCH. Cost: \$57,500.	139
IDR Inc. IDR Computer Module Single-board computer	A 386-based, IBM PC-AT compatible, single-board computer module for OEMs and VARs. Features 2M bytes of zero-wait-state memory, enhanced graphics display adapter, hard disk controller, floppy disk controller, two serial communications ports, one parallel printer port, and a real-time clock. No price given.	140
Macrotech International M1386S Satellite board	A 16-MHz 80386 satellite board for the S100 bus. Up to seven satellite boards possible in a system (more with a custom PAL). Features 1M byte of 32-bit dynamic memory and 1M byte of dual-ported, 32-bit, zero-wait-state, four-way interleaved, 100-ns dynamic RAM. Optional 80387 math coprocessor. Cost: \$2667.	141
Motorola MVME373 MAP controller board	An MC68000-based, VMEbus, Mini-MAP controller board, compatible with MAP physical connection requirements. Hosts an MC68824 Enhanced Token Bus Controller chip and Carrier-band RF Modem chip. Supports either the Mini-MAP three-layer stack or seven-layer MAP end-node protocol stack. Scheduled for fourth quarter. Cost: \$995.	142
Newer Technology DartCard Memory board	A modular RAM disk memory expansion system configurable with a variety of disk interfaces. Consists of a half- or full-height expansion card rack; plug-in DartCard memory boards of 4M, 16M, or 64M bytes; and choice of personality interfaces, including AT-bus multifunction, SCSI, and ESD1. No prices given.	143
Radius Inc. Accelerator 25 Processor card	A 25-MHz processor card for the Apple Macintosh SE. Based on the 32-bit Motorola 68020. Features high-speed logic with a write-through hardware cache of 32K bytes of static RAM. Works with Apple's single in-line memory modules. Optional floating-point coprocessor, the Motorola 68881. Cost: around \$2000.	144
Scientific Micro Systems SMS 1000 Model 38 Microcomputer	A microcomputer based on Digital Equipment's Q-bus architecture and compatible with the DEC LSI-11 computer. Includes a Winchester hard disk drive, an LSI-11/23 or LSI-11/73 CPU, four or eight serial ports, removable media, and a five-slot quad-height Q-bus backplane. Cost: \$3000 to \$17,000 based on configuration and quantity.	145
Waterworks Software RAM Lord RAM manager	A RAM management utility. Eliminates software conflicts and memory overload by having a PC swap TSR/RAM-resident packages in and out of expanded or extended memory or from hard disk or floppy disk space. Also eliminates duplicate hot keys. Cost: \$99.95.	146

Symposium on the Engineering of Computer-Based Medical Systems

June 8-10, 1988 • Minneapolis, Minnesota



Engineering in Medicine and Biology
Society of the IEEE



The Computer Society of the IEEE

UNIVERSITY OF MINNESOTA

The symposium will explore the full range of issues and problems that emerge from the process of engineering computer-based medical systems. Sessions will be held in the following areas:

- hardware architecture and applications
- reliability, fault tolerance, and design for testability
- software quality assurance and validity
- artificial intelligence and expert systems
- regulatory concerns and strategies
- chronobiology and closed-loop drug delivery systems
- software applications

In addition there will be four pre-conference tutorials:

- Medical Computing: The Big Picture
- Hardware Design for Reliability
- Software Safety
- Regulatory Issues and Standards

A WorldMed '88 Event

This symposium is part of WorldMed '88, the second in a biannual series of major conferences in Minnesota focusing on medical technology and the healthcare industry. Meeting concurrently with the IEEE symposium in the Minneapolis Hyatt Regency are conferences on (1) biomedical technology transfer and (2) competitive business strategies in the healthcare industry. Registration for one component of WorldMed '88 provides entrance to all three, and to a general floor.

WorldMed Keynote Speakers

- John Villforth, Director, Center for Medical Devices and Radiological Health, FDA
"Medical Devices and Regulatory Issues"
- Wilbur Gantz, President, Baxter Healthcare Corp.
"How Can Technology Meet the Needs of the Healthcare Market?"
- Werner Maly, Generalbevollmächtigter Direktor, Siemens Medical Engineering Group, Erlangen, West Germany
"Strategic Planning in the World Market for Medical Products"

Symposium Committees

EXECUTIVE COMMITTEE

Dr. John Long

Department of Surgery
University of Minnesota
2829 University Avenue S.E.
Minneapolis, MN 55414
(612) 627-4850

Dr. Troy Nagle

North Carolina State University
Raleigh, NC
(Representing IEEE CS)

Dr. Al Potvin

Eli Lilly & Company
Indianapolis, IN
(Representing IEEE EMBS)

PROGRAM CHAIRMEN

Dr. Tim Kriewall

Sarnes Inc./3M
Ann Arbor, MI

Dr. Max Cortner

Unisys
St. Paul, MN

Mr. Greg Sachs

Biomedicus
Eden Prairie, MN

COOPERATING ORGANIZATIONS

IEEE Healthcare Engineering
Policy Committee

Region 7 IEEE

Twin Cities Chapters of CS, EMBS
IEEE CS Technical Committee on
Computational Medicine

For More Information or to Register

Call or write the

Office of Continuing Medical Education
University of Minnesota
Box 202 UMHC
420 Delaware Street S.E.
Minneapolis, MN 55455
Telephone (612) 626-5525



THE COMPUTER SOCIETY

A member society of the Institute of Electrical and Electronics Engineers, Inc.

Executive Committee

President: Edward A. Parrish, Jr.*
Vanderbilt University
School of Engineering
Nashville, TN 37235
(615) 322-2762

President-Elect: Kenneth R. Anderson*

Vice Presidents
Education: Gerald L. Engel†
Technical Activities: Helen M. Wood (1st VP)*
Area Activities: Willis King†
Conferences and Tutorials: Joseph E. Urban (2nd VP)*
Membership and Information: Merlin G. Smith†
Publications: James H. Aylor†
Standards: Laurel Kaleda

Secretary: Duncan H. Lawrie
Treasurer: Barry W. Johnson†
Past President: Roy L. Russo*

Division V Director: Harriett Rigas†
Division VIII Director: H. Troy Nagle, Jr.†
Executive Director: T. Michael Elliott†

*voting member of the Board of Governors
†nonvoting member of the Board of Governors

Board of Governors

Term Ending 1988	Term Ending 1989
Mario Barbacci	Bill D. Carroll
Victor Basili	Lansing Hatfield
Paul L. Borrell	Duncan H. Lawrie
Lorraine M. Duvall	David Pessel
Michael Evangelist	Susan L. Rosenbaum
Allen L. Hankinson	Sallie V. Sheppard
Laurel Kaleda	Bruce D. Shriver
Ted Lewis	Harold S. Stone
Ming T. Liu	Akihiko Yamada
Earl E. Swartzlander, Jr.	Marshall C. Yovits

Next Board Meeting

8:30 a.m., June 17
Anaheim Sheraton, Anaheim, Calif.

Senior Staff

Executive Director: T. Michael Elliott
Editor and Publisher: True Seaborn
Director, Computer Society Press: Chip G. Stockton
Director, Conferences: Anne Marie Kelly
Director, Finance and Administration: Mary Ellen Curto

Offices of the Computer Society

Headquarters Office
1730 Massachusetts Ave. NW
Washington, DC 20036-1903
Phone: (202) 371-0101
Telex: 7108250437 IEEE COMPSO

Publications Office
10662 Los Vaqueros Circle
Los Alamitos, CA 90720
Membership and General Information: (714) 821-8380
Publications Orders: (800) 272-6657

European Office
13, Avenue de l'Aquilon
B-1200 Brussels, Belgium
Phone: 32 (2) 770-21-98
Telex: 25387 AVVALB

Purpose

The Computer Society strives to advance the theory and practice of computer science and engineering. It promotes the exchange of technical information among its 90,000 members around the world, and provides a wide range of services which are available to both members and nonmembers.

Membership

Members receive the highly acclaimed monthly magazine *Computer*, discounts on all society publications, discounts to attend conferences, and opportunities to serve in various capacities. Membership is open to members, associate members, and student members of the IEEE, and to non-IEEE members who qualify as affiliate members of the Computer Society.

Publications

Periodicals. The society publishes six magazines (*Computer*, *IEEE Computer Graphics and Applications*, *IEEE Design & Test of Computers*, *IEEE Expert*, *IEEE Micro*, *IEEE Software*) and three research publications (*IEEE Transactions on Computers*, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *IEEE Transactions on Software Engineering*).

Conference Proceedings, Tutorial Texts, Standards Documents. The society publishes more than 100 new titles every year.

Computer. Received by all society members, *Computer* is an authoritative, easy-to-read monthly magazine containing tutorial, survey, and in-depth technical articles across the breadth of the computer field. Departments contain general and Computer Society news, conference coverage and calendar, interviews, new product and book reviews, etc.

All publications are available to members, nonmembers, libraries, and organizations.

Activities

Chapters. Over 100 regular and over 100 student chapters around the world provide the opportunity to interact with local colleagues, hear experts discuss technical issues, and serve the local professional community.

Technical Committees. Over 30 TCs provide the opportunity to interact with peers in technical specialty areas, receive newsletters, conduct conferences, tutorials, etc.

Standards Working Groups. Draft standards are written by over 60 SWGs in all areas of computer technology; after approval via vote, they become IEEE standards used throughout the industrial world.

Conferences/Educational Activities. The society holds about 100 conferences each year around the world and sponsors many educational activities, including computing sciences accreditation.

European Office

This office processes Computer Society membership applications and handles publication orders. Payments are accepted by cheques in Belgian francs, British pounds sterling, German marks, Swiss francs, or US dollars, or by American Express, Eurocard, MasterCard, or Visa credit cards.

Ombudsman

Members experiencing problems — late magazines, membership status problems, no answer to complaints — may write to the ombudsman at the Publications Office.

Information

Use the Reader Service Card to obtain the following material:

- Membership information and application (RS #202)
- Publications catalog (proceedings, tutorials, standards) (RS #201)
- Periodicals subscription application/information for individuals (members, sister-society members, others) (RS #200)
- Periodicals subscription application/information for organizations (libraries, companies, etc.) (RS #199)
- List of awards and award nomination forms (RS #198)
- Technical committee list and membership application (RS #197)
- Directory of officers, board members, committee chairs, representatives, staff, chapters, standards working groups, etc. (RS #196)

Also see membership application in this magazine.

CONFERENCES

Editor: Edmund L. Gallizzi, Computer Science Dept., Eckerd College, St. Petersburg, FL 33733; (813) 864-8272; Comppmail, e.gallizzi

Innovator of timed Petri nets keynotes international workshop

C.M. Woodside, Carleton University

Carl Adam Petri, whose doctoral thesis 25 years ago proposed the model that now bears his name, delivered the keynote address at the International Workshop on Petri Nets and Performance Models. His talk, "Tools of General Net Theory," delved into the mathematics and the potential applications of ordinary Petri net models, providing a background for later discussions involving nets with time.

Timed Petri nets are emerging from current research as one of the basic approaches for predicting time delay and throughput performance in complex systems, including computer and communications systems. Ninety researchers from nine countries attended the 1987 workshop at the University of Wisconsin/Madison to discuss recent work in the area, including applications, ways to make the approach more powerful, and ways to combine it with better-known performance analysis methods like simulation and queueing analysis. It was the second meeting in a series that began in Turin, Italy, in 1985.

Petri's address dealt with three main topics: net operations, net morphisms, and net topologies.

The discussion on net operations first defined a fundamental operation called simplification that produces so-called "simple nets." Other participants presented further operations on these simple nets and demonstrated them to be useful in producing nets with desirable properties.

In his remarks on the concept of net morphisms, Petri characterized a net morphism as a mapping of a net to a net, which, as a result, preserves some important relations among the net components.

In his discussion on net topologies, Petri advocated an operational (that is, computerizable) approach to topology that produces a theory of concurrency and provides a new way to generate combinatorial models from continuous models.

Applications evolved as a major topic of presentations and discussion throughout the workshop. The applications

considered included

- communications protocols for local area networks,
- software performance, including parallel tasks with rendezvous,
- distributed file systems,
- clock synchronization,
- VLSI performance,
- multiprocessors and interconnection nets for them, and
- flexible manufacturing systems.

The applications used timed net models to find expected delay in complex sequences of actions; average throughput capacities of parallel computers, communications channels, or manufacturing systems; and mean times to failure, or average failure rates, in fault-tolerant designs. In some cases, the net models were also used for correctness analysis.

The practical use of Petri nets almost always requires a computational tool, and it was generally agreed that the present state of the tools limits the applications to small systems and analysis by experts. All the applications described related to small systems, although most of them were real (rather than paper systems) and dealt with real engineering design issues. The systems must be small because the tools require large amounts of space and time, both increasing at exponential rates with the number of elements in the system.

Although the models are easy to construct, expertise is required to find errors made during modeling. During the discussion, it was stated that the industry is looking forward to the development of methods for debugging net models. Despite these difficulties, several large companies in Europe and North America have begun to develop the expertise to use existing net tools and are using them in design calculations.

The search for ways to do performance calculations on larger systems, with large numbers of interacting processes or processors, led to discussion of ways to exploit symmetries in some systems as well as to approximate solutions. Based on various decompositions of systems, approximations were described

as modular-hierarchical, by time scale of events, and by functional layering. Several groups are concentrating in this direction. However, the researchers admittedly have little experience as yet with the accuracy or robustness of their approximations.

Another direction of current work is how to best exploit structural analysis of the net—using the extensive mathematical theory developed for ordinary (untimed) Petri nets—to get performance information. With this approach, workshop participants showed that it is possible to determine the existence of bottlenecks (which impose processing throughput limits).

Some computational tools were exhibited, including the "design" graphical tool from Meta Software, a company that also helped fund the workshop. Two user-defined tools created with the "design" software were shown; one is from MIT and the other is from the University of Wisconsin at Madison.

Some of the other tools featured in the discussions come from the Politecnico di Torino, Duke University, the University of California at Irvine, and the University of Michigan at Ann Arbor. Some tools have extensive graphical capability, including hierarchical structuring; some trace execution sequences; some do structural analysis as well as timings; and several include simulators. There were wall-to-wall acronyms (GTPN, GSPN, DTPN, XSPN, ESPN, etc.) for the different flavors of net models implemented in the tools. However, the two main variations use deterministic versus random times in describing events and intervals, and provide the two main "cultures" of the field.

The proceedings of the workshop are available from the Computer Society Press, 10662 Los Vaqueros Circle, Los Alamitos, CA 90720-2578; (800) CS-BOOKS. When ordering, specify catalog number 87TH0815-9.

The next workshop in the series will be held in Kyoto, Japan, in December 1989.

Copyright protection appeal, compiler demonstration highlight Micro 20

Gearold Johnson, Colorado State University

A call for a more precise software and firmware copyright protection definition, plus an on-line demonstration of a retargetable microcode compiler, highlighted a four-day Colorado Springs, Colo., conference.

The event was the 20th annual Workshop on Microprogramming, jointly sponsored by the Computer Society's Technical Committee on Microprogramming (TC Micro) and the ACM's Special Interest Group on Microprogramming (SIGMicro) in December. Gearold Johnson, outgoing SIGMicro chair, was chair of the workshop.

Opting against having a keynote speaker, Micro 20's organizers scheduled a presentation by Dennis Karjala, a law professor at Arizona State University who also has a PhD in electrical

engineering from the University of Illinois.

In his talk, entitled "Copyright, Computer Software, and The New Protectionism," he attacked the current wave of court cases concerning software and firmware and urged the adoption of a much more specific definition for copyright protection than most software developers are now claiming through litigation.

The discussion following the presentation was so lively that it had to be adjourned until later in the evening, when it continued for another 3½ hours.

Karjala's paper has been published in Vol. 28, No. 1, of *Jurimetrics: Journal of Law, Science and Technology*.

The retargetable microcode compiler demonstration repeated the demonstra-

tion that had taken place during the "great microcode compiler cook-off" competition part of the workshop.

Joe Linn of the Institute for Defense Analysis and the current TC Micro chair had put together a family of microarchitectures that was distributed as part of the workshop information packet. Two of the six family members were proposed for the competition: one for students and another for industrial-grade microcode compilers.

No student-developed compilers were submitted, but Lothar Nowak of the University of Kiel in West Germany demonstrated the institute's Mimola retargetable microcode compiler on the Linn architecture. The entry was declared the winner of the competition.

The software was demonstrated on

A retrospective on two decades of Microprogramming Workshops

Edmund L. Gallizzi

As with many areas of computer science, microprogramming has enjoyed rapid and continuous growth in knowledge. Much of this growth has been directly supported by the yearly Workshops on Microprogramming that have been conducted over the past two decades.

The workshops draw a small, specialized core group of attendees, as do many specialized workshops and conferences. The inspiration for many important ideas and for research has grown out of the discussions and arguments of small groups of participants at the various workshops.

The series was one of the first to be jointly sponsored by the Computer Society and the Association for Computing Machinery through the respective Technical Committee on Microprogramming (TC Micro) and the Special Interest Group on Microprogramming (SIGMicro).

Gearold Johnson, the SIGMicro chair from 1983 to 1987 and the chair of Micro 20, outlined the overall technical history of the workshop as starting with an interest in the hardware of microprogrammed systems.

This interest was engendered at Micro 1 in 1968 and continued strong until Micro 5 or 6 in the early 70s. It was the era of the Nanodata QM1, a general-purpose micro and nano-programmable host, and the B1700 with its compiler optimized microprogram and writable control store.

Johnson's outline described Micros 7 to 9 in the mid-70s as a time when the early "pioneering" high-level microprogramming languages

were presented. Of this era, Scott Davidson of AT&T, the TC Micro chair in 1983-84, said one of the goals of the research presented at the workshops was to promote the development of machine-independent, general-purpose, high-level microprogramming languages and systems.

Some of the pioneering high-level languages included S* by Subrata Dasgupta, now of the University of Southwestern Louisiana; EMPL by David DeWitt of the University of Wisconsin; MPL by Richard Eckhouse, now of MOCO, Inc.; STRUM by David Patterson of the University of California/Berkeley; and SIMPL by C.V. Ramamoorthy, also from UC Berkeley.

The machine-independent language development led to research in microcode compaction that provides a machine-dependent, optimization mechanism to generate microcode from the output of the machine-independent, high-level language compilers.

The debate surrounding the Reduced Instruction Set Computer (RISC) and the Complex Instruction Set Computer (CISC) was evident at the workshops from Micro 14 to Micro 16 in the early 80s, according to Johnson.

Part of the debate centered in the "Microcode Shoot-out," which has been a tradition since it was initiated several years ago by Will Tracz of IBM, the current SIGMicro chair.

According to Tracz, who has been the Shoot-out moderator each year, many of the participants "shot from the lip." Patterson, who has made

important contributions to microprogramming, was part of a heated Shoot-out over his new RISC design. Typically, RISC designs use no microprogramming, while the CISC systems are extensively microprogrammed.

The latest workshop topics have focused on the techniques of firmware engineering, as indicated by the Micro 13 keynote address on software engineering by William Riddle, now director of the Software Production Consortium. These software/firmware engineering techniques have succeeded in fostering the development of the production-quality microprogramming systems that were demonstrated at the latest workshop, Micro 20.

A particularly successful system that grew out of interaction at the workshop series is the Micro-C firmware development system of Robert Mueller of Colorado State University. According to Mueller, a paper that Preston Gurd delivered at a Micro 16 session when Mueller was chair sparked the concept for the system's development.

It's a production-quality microcode-development system based on the C programming language. The system is composed of an assembler, a stand-alone optimizer, an ANSI C compiler, a symbolic debugger, and a simulator. The target host architecture is defined by a common configuration language.

The current C language-based microprogramming systems have been very successful. According to Davidson, two important reasons for

an Apollo DN3000, loaned to the workshop by Apollo's Denver office. Copies of Linn's microarchitecture are available from the editor of SIGMicro's newsletter.

Nearly 90 researchers from 10 countries on four continents participated in the workshop, with two-thirds representing academic halls and the remaining attendees representing industry and government organizations.

A working meeting as opposed to a conference, the workshop had no parallel sessions; instead, attendees were encouraged to participate in discussions following the presentation of each paper and in evening discussion groups on active microprogramming topics.

As usual, the loudest discussions centered around the microprogramming "shoot-out" directed by Will Tracz of IBM. This is an annual event.

Speakers presented 22 papers at the workshop, including seven on micro-

code compaction and optimization techniques, three on pipeline architectures, three on microarchitecture tuning, three on real implementations, two on development systems, and one on applying formal methods to writing correct microcode.

Vicki Allan of Utah State University presented a very lucid overview of compaction as it relates to horizontal machines. Her presentation was judged the best of the workshop.

Nowak's paper, focusing on graph-based retargetable microcode compilation in the Mimola design system, presented the details of the process of retargeting and also provided an overview of the entire system. It was a very impressive piece of work.

Fred Homewood of Inmos Ltd., U.K., presented a fascinating paper on the use of software tools to map Occam processes to silicon. In this way, distributed systems of communicating processors are treated as though they are simple cells. The use of Occam as a higher level language greatly simplifies the process of building processor elements.

David Shepherd, also of Inmos Ltd., presented a paper on the use of mathematical logic and formal methods to write provably correct microcode. Researchers have discussed this issue for years; it was nice to finally see it in use. This focus within Inmos seems to be through the influence and direction of David May, the master architect of both Occam and the transputer.

Stephen Melvin of the University of California/Berkeley presented the results of a microcode-based tool analysis for tracing operating system events. The presentation included "real" data for the VAX 8600. It was an outstanding paper for those looking for performance-distribution data.

To close out the workshop, David Archer of Digital Equipment Corporation presented a paper on the microcode aspects of Digital's CVAX architecture.

Additional papers dealt with optimization issues, microcode for Lisp machines, pipeline architectures, and so on; each was important in its own way. In addition, TC Micro and SIGMicro conducted a joint evening board meeting.

This year's Micro 21 will be held in San Diego. Additional demonstrations, perhaps even the first student entry, are sought.

If all goes according to plan, Micro 22 in Dublin, Ireland, in 1989 will not only be the 22nd Workshop on Microprogramming but also the First International Workshop on Microprogramming and Microarchitecture.

Phoenix conference keynoter to focus on AI applications

John McCarthy of Stanford University will present the keynote address when he appears at the seventh annual Phoenix Conference on Computers and Communications in Scottsdale, Ariz., March 16-18. The title of his talk is "Current and Future Applications of Artificial Intelligence."

McCarthy has been a major contributor to the AI field since the earliest days of computing. His interest in AI began in 1949. His numerous accomplishments include developing the Lisp programming language in 1958, receiving the ACM's A.M. Turing Award in 1971 and the first Research Excellence Award of the International Joint Conference on Artificial Intelligence in 1985, and election to the National Academy of Engineering in 1987.

The conference will also include two days (March 17 and 18) of five concurrent tracks covering computers, communications, software, network technologies, and AI. A full day of tutorials will be presented March 16.

11th Testability Workshop slated

Focusing on current technology, the Computer Society will sponsor the 11th annual IEEE Workshop on Design for Testability April 19-22 in Vail, Colo.

Richard L. Lemke, general manager of computer-aided engineering for Tektronix, will keynote the workshop. His address is entitled "Linking Design and Test—Using Virtual Instruments to Explore Testability."

"The increasing complexity of both electronics products and test equipment creates the need for software that facilitates verification of design and physical prototypes," Lemke said. "Computer-aided test software works with both real and virtual instruments. Virtual instruments verify design integrity and testability. Real instruments verify the physical process."

Workshop Chair T.W. Williams of IBM said other key sessions of the workshop will be devoted to the emerging fields of biased random pattern self testing and delay testing techniques. Both areas are drawing a great deal of interest, Williams said.

Additional information may be obtained by contacting T.W. Williams, IBM Corp.—67A/02I, 6300 Diagonal Hwy, Boulder, CO 80301; (303) 924-7692.

this success are YACC and LEX, the compiler-generation tools of the Unix operating system. These tools have allowed the developers to focus on the microprogramming issues.

Maurice V. Wilkes, the developer of microprogramming and the leader of the group at the Mathematical Laboratory of Cambridge that built the first operational stored program computer, presented keynote addresses at Micros 10 and 15. At the latter session, Wilkes described the Microprogramming Repository. The University of Southwestern Louisiana facility was started by Bruce Shriver, who is with the IBM T.J. Watson Research Center and is the editor-in-chief of *Computer*.

During the last several workshops, from Micro 17 to Micro 20, the implications of the legal issues of microcode copyright have been an important area of discussion.

The current state of microprogramming technology is the subject of a book entitled *Methods of Microprogramming and Firmware Engineering* edited by Stanley Habib, one of the early SIGMicro chairs. The book will be available this June.

Although the ideas microprogramming generates are important in themselves, and also pervade many other areas of computer architecture, a continuing question arises at the workshops. The question is: "What is microprogramming?" It's a question that persists not because of a lack of answers but because of an abundance of answers.

CALENDAR

March 1988

1988 International Development Center Conference (DCI), Mar. 13-16, Orlando, Fla. Contact Development Center Institute, PO 44087, Indianapolis, IN 46244-0087; (317) 846-2753.

Sixth National Conference on Ada Technology, Mar. 14-17, Washington, DC. Contact Al Rodriguez, US Army Communications-Electronics Command, AMSEL-RD-LC-ASST-1A, Fort Monmouth, NJ 07703; (201) 532-4725.

 **CAIA 88, Fourth International Conference on Artificial Intelligence Applications, Mar. 14-18, San Diego.** Contact Jim Miller, MCC, Human Interface Program, 3500 W. Balcones Center Dr., Austin, TX 78720; (512) 338-3342, or CAIA 88, Computer Society of the IEEE, 1730 Massachusetts Ave., NW, Washington, DC 20036-1903; (202) 371-1013.

International Conference Extending Data Base Technology (AICA, AFCET, BCS), Mar. 14-18, Venice, Italy. Contact Joachim W. Schmidt, Universitat Frankfurt, Fachbereich Informatik, Dantestr. 9, D6000 Frankfurt 1, West Germany; phone (49) 69-798-8101.

Securicom 88, Sixth Worldwide Congress on Computer and Communications Security and Protection, Mar. 15-17, Paris. Contact Securicom 88, 8, rue de la Michodiere, 75002 Paris, France; phone (33) 1-47-424-100.

 **PCCC 88, Seventh IEEE Phoenix Conference on Computers and Communications, Mar. 16-18, Scottsdale, Ariz.** Contact PCCC 88, c/o Carl Ryan, Motorola GEG, 2501 S. Price Rd., Chandler, AZ 85248-2899; (602) 732-3074.

21st Simulation Symposium (SCS, IMACS), Mar. 16-18, Tampa, Fla. Contact Alfred Jones, Computer Science Dept., Florida Atlantic University, Boca Raton, FL 33431; (305) 393-3675.

Western Educational Computing Workshops (CECC), Mar. 17-18, Norwalk, Calif. Contact Alexia Devlin, San Francisco State University, Accounting Data, NADM-358, 1600 Holloway Ave., San Francisco, CA 94132.

11th PACS Computer Festival, Mar. 19, Philadelphia, Pa. Contact Stephen A. Longo, Philadelphia Area Computer Society, c/o La Salle University, Philadelphia, PA 19141; (215) 951-1255.

 **20th Southeastern Symposium on System Theory, Mar. 20-22, Charlotte,**

N.C. Contact William A. Smith, Dept. of Electrical Engineering, University of North Carolina, Charlotte, NC 28223; (704) 547-4142.

NCGA 88, Ninth Conference and Exposition on Computer Graphics Applications, Mar. 20-24, Anaheim, Calif. Contact NCGA 88, National Computer Graphics Assoc., 2722 Merrilee Dr., Suite 200, Fairfax, VA 22031.

 **Compstan 88, Computer Standards Conference, Mar. 21-23, Arlington, Va.** Contact James Hall, US Dept. of Commerce, National Bureau of Standards, Technology Bldg. 225, Rm. B266, Gaithersburg, MD 20899; (301) 975-3273.

RIAO 88, User-Oriented Content-Based Text and Image Handling Conference (AFIPS), Mar. 21-24, Cambridge, Mass. Contact RIAO 88 Conference Service Office, Massachusetts Institute of Technology, Bldg. 7, Rm. 111, Cambridge, MA 02139; or CID, 36 bis, rue Ballu, 75009 Paris, France.

 **Sixth IEEE VLSI Test Workshop, Mar. 22-23, Atlantic City, N.J.** Contact Wesley E. Radcliffe, IBM East Fishkill, Dept. 277, Bldg. 321-5E1, Hopewell Junction, NY 12533; (914) 894-4346.

AAAI Spring Symposium, Mar. 22-24, Palo Alto, Calif. Contact American Assoc. for Artificial Intelligence, 445 Burgess Dr., Menlo Park, CA 94025-3496; (415) 328-3123.

BANKAI 88 (SWIFT), Mar. 22-24, London. Contact Jean Steinier, Society for Worldwide Interbank Financial Telecommunications, PO 2005, Culpeper, VA 22701; (703) 829-1300.

CAE India 88, India Computer Graphics International Conference and Exhibition, Mar. 22-25, New Delhi, India. Contact Tara S. Ganguli, Technology and Research Associates, 5 Lindsay St., Calcutta 700087, India; phone (033) 299-420.

 **COIS 88, Conference on Office Information Systems (ACM), Mar. 23-25, Palo Alto, Calif.** Contact Robert Allen, 2A-367, Bell Communications Research, Morristown, NJ 07960; (201) 829-4315.

 **Extending Database Technology (IASI, INRIA), Mar. 23-25, Venice, Italy.** Contact Stefano Ceri, Politecnico di Milano, Dipart. di Elektronika, Piazza Leonardo da Vinci 32, 20133 Milano, Italy; phone 39 (02) 236-7241.

 **Built-In Self-Test Workshop, Mar. 23-25, Charleston, S.C.** Contact Richard Sedmak, Self-Test Services, 6 Lin-

denwold Ter., Ambler, PA 19002; (215) 628-9700 or 2106.

International Congress on CIM Databases, Mar. 27-29, Cambridge, Mass. Contact Kim Takita, 824 Boylston St., Chestnut Hill, MA 02167; (617) 232-8080.

Software Publishers Assoc. 1988 Spring Symposium, Mar. 27-30, Berkeley, Calif. Contact SPA Conference Registration, 1101 Connecticut Ave. NW, Suite 901, Washington, DC 20036; (202) 452-1600.

Fourth International Conference on Pattern Recognition (BPRA, IAPR), Mar. 28-30, Cambridge, England. Contact J. Kittler, Dept. of Electronic and Electrical Engineering, University of Surrey, Guildford GU2 5XH, England, UK.

 **Infocom 88, Conference on Computer Communications, Mar. 28-31, New Orleans.** Contact Infocom 88, c/o Ron Rutledge, Martin Marietta Energy Systems, MS 271, Bldg. 4500N, Oak Ridge National Laboratories, PO X, Oak Ridge, TN 37831; (615) 625-7643.

April 1988

 **Applications of Artificial Intelligence VI, Apr. 4-6, Orlando, Fla.** Contact Mohan M. Trivedi, University of Tennessee, ECE, Ferris Hall, Knoxville, TN 37996-2100; (615) 974-5450.

Conference on the Human Dimension in Artificial Intelligence, Apr. 6-9, Lexington, Ky. Contact Engineering Continuing Education, 223 Transportation Research Bldg., University of Kentucky, Lexington, KY 40506-0043; (606) 257-4295.

 **Computer Networking Symposium, Apr. 11-13, Washington, DC.** Contact George K. Chang, Bell Communications and Research, 6 Colbert Pl., Piscataway, NJ 08854; (201) 699-3879.

 **CompEuro 88, Apr. 11-15, Brussels.** Contact Jacques Tiberghien, Vrije Universiteit Brussel, Pleinlaan 2, B 1050 Brussels, Belgium; phone 32 (02) 641-2905.

 **ICSE 88, 10th International Conference on Software Engineering (ACM), Apr. 11-15, Raffles City, Singapore.** Contact Tan Chin Nam or Lim Swee Say, National Computer Board, 71 Science Park Dr., NCB Building, Singapore 0511; phone (65) 772-0405.

PC 88, 17th International Programmable Controllers Conference (ESD, SMI), Apr.

12-14, Detroit. Contact IPC 88, 100 Farnsworth, Detroit, MI 48202; (800) 457-9504.

⊕ **VHDL Users' Workshop, Apr. 17-20**, Charlottesville, Va. Contact Ron Waxman, University of Virginia, Thornton Hall, Charlottesville, VA 22901.

Information Systems Conference (ASM), Apr. 17-20, San Diego, Calif. Contact Assoc. for Systems Management, 24587 Bagley Rd., Cleveland, OH 44138; (216) 243-6900.

⊕ **1988 IEEE Symposium on Security and Privacy, Apr. 18-21**, Oakland, Calif. Contact Dave Bailey, US Dept. of Energy, Production Operations Division/CIM Office, PO 5400, Albuquerque, NM 87115; (505) 846-4600.

Eastern Simulation Conference (SCS), Apr. 18-21, Orlando, Fla. Contact Society for Computer Simulation, PO 17900, San Diego, CA 92117; (619) 277-3888.

⊕ **Workshop on Factory Communications, Apr. 19-20**, Gaithersburg, Md. Contact Alfred C. Weaver, Dept. of Computer Science, University of Virginia, Charlottesville, VA 22903; (804) 979-7529.

⊕ **11th IEEE Workshop on Design for Testability, Apr. 19-22**, Vail, Colo. Contact T.W. Williams, IBM Corp., PO 1900, Dept. 67A/021, Boulder, CO 80301-9191; (303) 924-7692.

⊕ **Second International Conference on Expert Database Systems, Apr. 25-28**, Tysons Corner, Va. Contact Edgar H. Sibley, George Mason University, ICSE Dept., 4400 University Dr., Fairfax, VA 22030.

Working Conference on Parallel Processing (IFIP), Apr. 25-27, Pisa, Italy. Contact Michel Cosnard, Laoratoire TIM3, Institut National Polytechnique de Grenoble, 46 Avenue Felix Viallet, 38031 Grenoble Cedex, France; phone (33) 7643-3726.

IEEE International Conference on Robotics and Automation, Apr. 25-29, Philadelphia. Contact Harry Hayman, 738 Whitaker Ter., Silver Spring, MD 20901; (301) 434-1990.

Sococo 88, Symposium on Software for Computer Control (IFAC, IFIP), Apr. 26-28, Johannesburg, South Africa. Contact IFIP, 3 rue de Marche, CH-1204, Geneva, Switzerland.

⊕ **Second Parallel Processing Symposium (California State University at Fullerton), Apr. 28-30**, Fullerton, Calif. Contact Larry Canter, 1619, N. Hale, Fullerton, CA 92631; (714) 738-3414.

May 1988

Machine Vision Hands-On Workshop, May 2-3, Atlanta. Contact Constantin Soulakos,

Georgia Tech Research Institute, O'Keefe Bldg, Rm. 40, Atlanta, GA 30332; (404) 894-3412.

AutoCAD Expo 88, May 2-5, Chicago. Contact Autodesk, Inc., 2320 Marinship Way, Sausalito, CA 94965; (415) 332-2344.

Second Military Computing Conference, May 3-5, Anaheim, Calif. Contact Military Computing Conference, PO 428, Los Altos, CA 94023; (415) 494-2800.

Artificial Intelligence and Advanced Computer Technology Conference and Exhibition (SCS), May 4-6, Long Beach, Calif. Contact Tower Conference Management Co., 800 Roosevelt Rd., Glen Ellyn, IL 60137-5835; (312) 469-3373.

19th Pittsburgh Conference on Modeling and Simulation (IEEE, ISA, SCS), May 5-6, Pittsburgh. Contact William G. Vogt or Marlin H. Mickle, Modeling and Simulation Conference, 348 Benedum Engineering Hall, University of Pittsburgh, Pittsburgh, PA 15261.

Workshop on Parallel and Distributed Debugging (ACM), May 5-6, Madison, Wis. Contact Bart Miller, Computer Sciences Dept., University of Wisconsin, 1210 W. Dayton St., Madison, WI 53706; (608) 263-3378.

⊕ **Fourth International Software Process Workshop, May 11-13**, Devon, England. Contact Leon Osterweil, Computer Science Dept., University of Colorado, Campus Box 430, Boulder, CO 80309; (303) 492-8787.

⊕ **Fifth Workshop on Real-Time Operating Systems, May 12-13**, Washington, DC. Contact John A. Stankovic, Dept. of Computer and Information Science, University of Massachusetts, Amherst, MA 01003; (413) 545-0720.

Software Maintenance Assoc. Conference, May 15-18, Chicago. Contact Suzanne Grenoble, SMA, Box 391432, Mountain View, CA 94039; (408) 730-1132.

CHI 88, Conference on Human Factors in Computing Systems (ACM), May 15-19, Washington, DC. Contact Gail A. Chumura, 5214 Monroe Dr., Springfield, VA 22151; (703) 750-9401.

Second Symposium on Space C³ (AFCEA), May 16-19, Annapolis, Md. Contact Randy Crawford/DQ, IIT Research Institute, 185 Admiral Cochrane Dr., Annapolis, MD 21401; (301) 224-2295.

NATO Advanced Research Workshop on Highly Redundant Sensing in Robotic Systems, May 16-21, Tuscany, Italy. Contact Julius Tou, Center for Information Research, University of Florida, Gainesville, FL 32611; (904) 335-8018.

CIPS Congress 88, May 18-20, Toronto, Canada. Contact Canadian Information Processing Society, 243 College St., 5th

Floor, Toronto, Ontario, M5T 2Y1; phone (416) 593-4040.

41st SPSE Conference, May 22-27, Washington, DC. Contact Society for Imaging Science and Technology, 7003 Kilworth Lane, Springfield, VA 22151.

⊕ **Third International IEEE Conference on Ada Applications and Environments, May 23-26**, Manchester, N.H. Contact Derek S. Morris, Dept. of Electrical Engineering and Computer Science, Stevens Institute of Technology, Hoboken, NJ 07730; (210) 420-5606.

SID 88, May 23-27, Anaheim, Calif. Contact James N. Price, Naval Ocean Systems Center, Attn.: Code 713, San Diego, CA 92152; (619) 225-2665.

⊕ **18th International Symposium on Multiple-Valued Logic, May 24-26**, Madrid, Spain. Contact Enric Trillas, Investigaciones Cientificas, Serrano 117, 28006-Madrid, Spain; phone 34 (91) 621-6264.

CG International 88 (CGS, BCS), May 24-27, Geneva. Contact N. Magnenat-Thalmann, MIRALab HEC, 5255 Decelles, Montreal, Canada H3T 1V6.

SIGMetrics Conference on Measurement and Modeling of Computer Systems (ACM), May 24-27, Santa Fe, N.M. Contact Connie U. Smith, L and S Computer Technology, 1114 Buckman Rd., Santa Fe, NM 87501; (505) 988-3811.

⊕ **International Conference on Systolic Arrays, May 25-27**, San Diego. Contact Keith Bromley, Code 741-T, Naval Ocean Systems Center, 271 Catalina St., San Diego, CA 92152; (619) 225-7008.

International Workshop on Artificial Intelligence for Industrial Applications (IEEE, SICE), May 25-27, Hitachi, Japan. Contact Takao Sasayama, Technology Planning Group, Hitachi America Ltd., 50 Prospect Ave., Tarrytown, NY 10591-4698; (914) 332-5800, or Kotaro Hirawawa, 10th Dept., Hitachi Research Laboratory, Hitachi, Ltd., 4026 Kuji-cho Hitachi-shi, 319-12, Japan; (02) 94-52-5111.

15th IFAC Workshop on Real-Time Programming, May 25-27, Valencia, Spain. Contact WRTP 88, Grupo de Informatica Industrial—DSIC/DISCA, Universidad Politecnica de Valencia, PO 22012, E-46071 Valencia, Spain; phone (34) 6-360-4041.

Eurocrypt 88, Workshop on the Theory and Application of Cryptologic Techniques (IACR), May 25-27, Davos, Switzerland. Contact Paul Schobi, Ltd., Althardstr. 70, CH-8105 Regensdorf, Switzerland.

⊕ **15th International Symposium on Computer Architecture (ACM), May 30-June 3**, Honolulu. Contact H.J. Siegel, Supercomputing Research Center, 4380 Forbes Blvd., Lanham, MD 20706; (301) 731-3700.

CALL FOR PAPERS

Call for papers for Computer

Computer seeks articles for the special January 1989 theme issue on **Real Machines: Design Choices/Engineering Tradeoffs**.

Six copies of each complete manuscript should be submitted by **May 15, 1988** to Yale Patt, *Computer* Guest Editor, College of Engineering, Dept. of Electrical Engineering and Computer Science, 573 Evans Hall, University of California at Berkeley, Berkeley, CA 94720.

IEEE Software: Articles are sought for a special issue on software validation and verification describing the emerging technologies and the underpinnings of this proven technology. Submit article to Dolores Wallace, Guest Editor, National Bureau of Standards, Bldg. B266, Gaithersburg, MD 20899; (301) 975-3340, or to Ted G. Lewis, Editor-in-Chief, *IEEE Software*, c/o Computer Science Dept., Oregon State University, Corvallis, OR 97331; (503) 754-2744; CSnet lewis@oregon-state; Compmail + t.lewis.

IEEE Software: Articles are sought for a special issue that show how working systems use rapid-prototyping technology, what issues remain, and how rapid prototyping interacts with other areas. Submit paper to Giorgio Bruno, Guest Editor, Dip. Automatica e Informatica, Politecnico di Torino, Corso Duca degli Abruzzi 24, 10129 Torino, Italy; phone 39 (11) 556-7003, or to Ted G. Lewis, Editor-in-Chief, *IEEE Software*, c/o Computer Science Dept., Oregon State University, Corvallis, OR 97331; (503) 754-2744; CSnet lewis@oregon-state; Compmail + t.lewis.

IEEE Expert is seeking materials for publication. Submit articles on technology and AI applications to David Pessel, Editor-in-Chief, BP America, 4440 Warrensville Center Rd., Cleveland, OH 44128; reports on conferences, short subjects and papers on PCs, products, and resources to Henry Ayling, Managing Editor, *IEEE Expert*, 10662 Los Vaqueros Cir., Los Alamitos, CA 90720; and book reviews to K.S. Shankar, Associate Editor, Federal Systems Division, IBM Corp., 3700 Bay Area Blvd., Houston, TX 77058.

Technical Committee on Computer Education, Computer Society of the IEEE: Contributions are welcomed for the TCCE newsletter, a forum for the exchange of ideas among persons interested in computer education or computers in education. Submit news items, short articles, and correspondence to Helen Hays, Dept. of Computer Science, Southeast Missouri State University, Cape Girardeau, MO 63701; (314) 651-2244.

IEEE Transactions on Reliability: Special issue on reliability of parallel and distributed computing networks scheduled for publication in late 1988. Submit contributions to Dharma P. Agrawal or Suresh Rai, Dept. of Electrical and Computer Engineering, PO 7911, North Carolina State University, Raleigh, NC 27695-7911; (919) 737-2336.

Supercomputing 88: Nov. 14-18, 1988, Orlando, Fla. Submit paper by **Mar. 14, 1988** to Stephen F. Lundstrom, ERL 455, Stanford University, Stanford, CA 94305; (415) 723-0140.

Summer Computer Simulation Conference (SCS): July 25-27, 1988, Seattle. Submit complete manuscript by **Mar. 15, 1988** to Charles A. Pratt, PO 17900, San Diego, CA 92117; (619) 277-3888.

Eighth International Conference in Computer Science: July 4-8, 1988, Santiago, Chile. Submit abstract by **Mar. 15, 1988** and final paper by **June 1, 1988** to Alberto O. Mendelzon, CSRI, University of Toronto, 10 King's College Rd., Toronto, Canada M5S 1A4.

OOPSLA 88, Conference on Object-Oriented Programming: Systems, Languages, and Applications (ACM): Sept. 25-29, 1988, San Diego. Submit paper by **Mar. 15, 1988** to Kurt Schmucker, OOPSLA 88, c/o Apple Computer, 5950 Symphony Woods Rd., Suite 410, Columbia, MD 21044.

International Software for Strategic Systems Conference: Oct. 25-26, 1988, Huntsville, Ala. Submit abstract by **Mar. 15, 1988** to Wayne Smith, 635 Discovery Dr., Huntsville, AL 35806; (205) 721-1941.

GOMAC 88, Government Microcircuit Applications Conference: Nov. 8-10, 1988, Las Vegas, Nev. Submit summary by **Mar. 16, 1988** to Jay Morreale, Palisades Institute for Research Services, G-88, 201 Varick St., Rm. 1140, New York, NY 10014.

CSM 88, Conference on Software Maintenance—1988 (DPMA. NBS): Oct. 24-27, Phoenix, Ariz. Submit paper by

Mar. 18, 1988 to Wilma M. Osborne, National Bureau of Standards, Bldg. 225, Rm. B266, Gaithersburg, MD 20899; (301) 975-3339.

IEEE Software and IEEE Expert: November 1988. Papers are sought for special issues on expert systems applied to software engineering. Submit manuscript by **Apr. 1, 1988** to Murat Tanik, Dept. of Computer Science and Engineering, Southern Methodist University, Dallas, TX 75275-0122; (214) 692-3083, ext. 2854.

Seventh Symposium on Reliable Distributed Systems: Oct. 10-12, 1988, Columbus, Ohio. Submit manuscript by **Apr. 1, 1988** to Jane W.S. Liu, Dept. of Computer Science, University of Illinois, 1304 W. Springfield Ave., Urbana, IL 61801-2987; (217) 333-0135.

ESIG 88, Fourth Expert Systems in Government Conference: Oct. 17-21, 1988, Washington, DC. Submit paper by **Apr. 1, 1988** to ESIG 88, MS W418, Mitre Corp., 7525 Colshire Dr., McLean, VA, 22102.

LFA 88, Sixth IEEE Workshop on Languages for Automation: Aug. 29-31, 1988, College Park, Md. Submit paper by **Apr. 1, 1988** to Panos A. Ligomenides, Electrical Engineering Dept., University of Maryland, College Park, MD 20742.

CCC 89, Second Hungarian Custom Circuit Conference (MATE): May 10-12, 1989, Szeged, Hungary. Submit paper by **Apr. 1, 1988** to MATE Secretariat, 1055 Budapest, Kossuth L. tér 6-8, Hungary; phone (1) 531-406.

ICNN 88, IEEE 1988 International Conference on Neural Networks: July 24-27, 1988, San Diego. Submit paper by **Apr. 1, 1988** to Nomi Feldman, IEEE ICNN 88, 3770 Tansy St., San Diego, CA 92121; (619) 453-6222.

International Workshop on Defect and Fault Tolerance in VLSI Systems: Oct. 6-7, 1988, Springfield, Mass. Submit extended summary by **Apr. 5, 1988** to Israel Koren, Dept. of Electrical and Computer Engineering, University of Massachusetts, Amherst, MA 01003; (413) 545-2643.

CASE 88, Second International Workshop on Computer-Aided Software Engineering (ACM): July 12-15, 1988, Cambridge, Mass. Submit position paper by **Apr. 11, 1988** to Pamela Meyer, CASE 88, c/o Index Technology Corp., One Main St., Cambridge, MA 02142; (617) 494-8200, ext. 1988.

1988 IEEE Workshop on Visual Languages: Oct. 10-12, 1988, Pittsburgh.

Submit paper by **Apr. 15, 1988** to Alfs T. Berziss, Computer Science Dept., Faculty of Arts and Sciences, University of Pittsburgh, 322 Alumni Hall, Pittsburgh, PA 15260; or (in Europe) to Stefano Levis, Dipartimento di Matematica, Università di Roma, Piazzale A. Moro, 00185, Rome, Italy.

ICCAD 88, IEEE International Conference on Computer-Aided Design (ACM): Nov. 7-10, 1988, Santa Clara, Calif. Submit paper by **Apr. 29, 1988** to ICCAD 88 Secretary, Electrical and Computer Engineering Dept., Carnegie Mellon University, Pittsburgh, PA 15213; (412) 268-3546.

13th Conference on Local Computer Networks: Oct. 10-12, 1988, Minneapolis, Minn. Submit paper by **May 1, 1988** to Larry Green, Advanced Computer Communication, 720 Santa Barbara St., Santa Barbara, CA 93101; (805) 963-9431.

Ninth Real-Time Systems Symposium: Dec. 6-8, 1988, Huntsville, Ala. Submit paper by **May 1, 1988** to Al K. Mok, Dept. of Computer Science, TAY 3-140C, University of Texas/Austin, Austin, TX 78712; (512) 471-9542.

International Symposium on Databases in Parallel and Distributed Systems: Dec. 5-7, 1988, Austin, Texas. Submit manuscript by **May 1, 1988** to Won Kim, 3500 Balcones Center Dr., Austin, TX 78759; (512) 338-3439.

ICCV 88, Second International Conference on Computer Vision: Dec. 5-8, 1988, Tarpon Springs, Fla. Submit paper by **May 15, 1988** to Ruzena Bajcsy, University of Pennsylvania, Dept. of Computer and Information Science, 200 S. 33rd St., Philadelphia, PA 19104-6389; (215) 898-6222.

IEEE Software: January 1989. Articles are sought on human-computer interfaces, human factors, and user issues. Submit articles by **June 1, 1988** to Virginia Hix, Guest Editor, Computer Science Dept., Virginia Polytechnic State University, Blacksburg, VA 24061; (703) 961-4857, or Ted G. Lewis, Editor-in-Chief, *IEEE Software* c/o Computer Science Dept., Oregon State University, Corvallis, OR 97331; (503) 754-2744; CSnet lewis@oregon-state; Compmail + t.lewis.

1988 International Computer Science Conference: Dec. 19-21, 1988, Hong Kong. Submit paper by **June 15, 1988** to Jean-Louis Lassez, IBM T.J. Watson Research Center, PO 218, Yorktown Heights, NY 10598, or Francis Y.L. Chin, Center of Computer Studies and Application, University of Hong Kong, Hong Kong.

Proceedings of the IEEE: January 1990. A special issue on state-of-the-art surveys on supercomputing technology is planned. Submit paper or proposal by **June 30, 1988** to Tse-yun Feng or A.R. Hurson, E.E. East Bldg., Pennsylvania State University, University Park, PA 16802; (814) 863-1469 or 1187.

CAREER OPPORTUNITIES

RATES: \$12.00 per line, \$120 minimum charge (up to ten lines). Average six typeset words per line, nine lines per column inch. Add \$10 for box number. Send copy at least one month prior to publication to: Heidi Rex or Marian Tibayan, Classified Advertising, *COMPUTER Magazine*, 10662 Los Vaqueros Circle, Los Alamitos, CA 90720; (714) 821-8380.

In order to conform to the Age Discrimination in Employment Act and to discourage age discrimination, *COMPUTER* may reject any advertisement containing any of these phrases or similar ones: "...recent college grads...", "...1-4 years maximum experience...", "...up to 5 years experience...", or "...10 years maximum experience." *COMPUTER* reserves the right to append to any advertisement, without specific notice to the advertiser, "Experience ranges are suggested minimum requirements, not maximums." *COMPUTER* assumes that, since advertisers have been notified of this policy in advance, they agree that any experience requirements, whether stated as ranges or otherwise, will be construed by the reader as minimum requirements only.

THE UNIVERSITY OF TENNESSEE SPACE INSTITUTE Faculty Positions in Computer Science

Applications are invited for tenure track positions at all levels. A Ph.D. in computer science or a closely related area and a commitment to research and teaching are required. Candidates from all areas of computer science will be considered; however, preference will be given to candidates with expertise in artificial intelligence or robotics. UTSI offers M.S. and Ph.D. degrees in computer science with emphasis on applied artificial intelligence, expert systems and robotics.

UTSI is a multidisciplinary graduate institute offering degree programs and research in engineering, physics, applied mathematics and computer science. Emphasis is placed on research and graduate-level instruction. In addition to a VAX/780 and VAX/785, two Symbolics 3600 Lisp machines are available for research as well as linkages to computers at The University of Tennessee in Knoxville and supercomputers at various locations.

Rank and salaries for these positions are open and commensurate with qualifications. Fringe benefits include group life and medical insurance, TIAA/CREF, and reduced tuition for dependents. UTSI occupies a scenic 365 acre lakeshore campus.

Send a detailed resume and names of three references to: Professor Moonis Ali, Chairman, Computer Science Committee, The University of Tennessee Space Institute, Tullahoma, TN 37388. (phone: (615) 455-0631, ext. 283).

UTSI is an AA/EEO employer

CONCORDIA UNIVERSITY Department of Electrical and Computer Engineering

The department invites applications for tenure track faculty positions at the Assistant, Associate or Full Professor levels in all areas of Computer Engineering. Our primary interest is in candidates with Computer Architecture background, particularly in the field of parallel or distributed processing. Candidates should have a strong interest in research as well as commitment to teaching at the undergraduate levels in Computer Hardware related courses. Medium of instruction is English. An earned doctorate is required. Degrees in Computer Engineering are of special interest. Salary and benefits are extremely attractive and negotiable. The department provides an excellent environment and opportunity for growth with established research groups in many areas. The department and the university have excellent computer facilities, modern teaching and research labs and support staff. Interested candidates should submit resumes with the names of three referees to: Jeremiah H. Hayes, Chairman, Electrical and Computer Engineering Department, CONCORDIA UNIVERSITY, 1455 de Maisonneuve Blvd. W., Montreal, Quebec, Canada H3G 1M8. In accordance with Canadian immigration requirements, this advertisement is directed to Canadian citizens and permanent residents.

UNIVERSITY OF MISSOURI—KANSAS CITY

And the beat goes on... In 1987 more faculty and staff were added, our new space was remodelled again to accommodate our continuing growth, a Center for Advanced Technology in Telecommunications and Computer Networking is being established within our unit, courses and degree programs are being televised and research equipment is being added to our labs. Telecommunications, Computer Networking, Artificial Intelligence and Software Studies are the four groupings of our research and instructional programs. If your work is in one of these areas and you are looking for a stimulating environment, please contact us.

Nine- or twelve-month appointments at all ranks are available at industry competitive salaries. Visiting and post-doctoral appointments are also open. Research productivity is required so time for research is guaranteed and increased as your research performance increases. Through the Center for Advanced Technology, substantial support for faculty and graduate student research in telecommunications and computer networking will be generated. Perhaps you are one of the few outstanding people we will be adding to our program this year.

Call or write: Dr. Richard G. Hetherington, Director, Computer Science, University of Missouri—Kansas City, 4747 Troost, Room 207, Kansas City, MO 64110. (816) 276-1193 (by April 15 for Fall semester appt.; by October 15 for Spring semester appt.) UMKC is an Equal Opportunity/Affirmative Action Employer.

UNIVERSITY OF HONG KONG
Lecturer in Computer Studies

Applications are invited for a Lectureship in the Centre of Computer Studies and Applications. Candidates should have a higher degree in Computer Science, Computer Engineering or Information Systems, and a strong interest in both teaching and research. Preference will be given to applicants with strong interest in research and development. Consideration may be given to applications for appointment on a short-term basis (please specify period). A certain amount of outside practice is permitted.

Annual salary (superannuable) is on an 11-point scale: HK\$188,040—314,340 (US \$1 = HK\$7.80 as at December 21 1987). Starting salary will depend on qualifications and experience. At current rates, salaries tax will not exceed 16½% of gross income. Children's education allowances, leave, and medical benefits are provided; housing or tenancy allowances are also provided in most cases at a charge of 7½% of salary.

Further particulars and application forms may be obtained from the Secretary General, Association of Commonwealth Universities (Appts), 36 Gordon Square, London WC1H 0PF, or from the Appointments Unit, Registry, University of Hong Kong, Hong Kong. Closes 31 March 1988.

UNIVERSITY OF WISCONSIN—MADISON
Faculty Positions

The Department of Electrical and Computer Engineering invites applications for tenure and tenure-track positions. A Ph.D. degree is required, and successful candidates are expected to participate in both teaching and research activities. Applicants in all areas of computer engineering are invited to apply, but the following areas are of special interest: computer architecture, computer networks, VLSI and computer-aided design, microprocessor and minicomputer applications, real-time control and instrumentation applications, and engineering applications of artificial intelligence. Rank and salary will be commensurate with qualifications and experience. Send resume and names of three references to J. Leon Shohet, Chairman, Department of Electrical and Computer Engineering, University of Wisconsin—Madison, 1415 Johnson Drive, Madison, WI 53706, an equal opportunity/affirmative action employer.

UNIVERSITY OF WISCONSIN—MILWAUKEE
Department of Electrical Engineering and
Computer Science

The Department expects to have Computer Science faculty openings. Candidates should have a Ph.D. in Computer Science or a closely related discipline. A strong commitment to research and teaching is expected. Senior candidates should have an excellent research record. Areas of special interest are: artificial intelligence, networks, compilers, software engineering, and theory.

The Department's current research strengths include theory, architecture, parallel computation, data-security and databases. The University campus is located near the shores of Lake Michigan and is close to pleasant and beautiful residential areas. Interested persons should send a resume with three references to:

Professor K. Vairavan
Chairman—Computer Science
Dept. of Elec. Eng. and Comp. Sci.
UW—Milwaukee
P.O. Box 784
Milwaukee, WI 53201

WORCESTER POLYTECHNIC INSTITUTE

The Computer Science Department invites applications for tenure track faculty positions at all levels from candidates in all areas of specialization. Candidates should have a Ph.D. in Computer Science and a strong interest in both research and teaching.

Worcester Polytechnic Institute emphasizes quality in the undergraduate learning experience and is committed to an innovative project-oriented teaching environment. The quality of the undergraduate computer science degree has been recognized by the recent granting of accreditation by the Computer Sciences Accreditation Board.

The current goal of the Institute is to enhance our graduate program and improve research activities. The department seeks qualified candidates who will help us achieve these objectives. WPI is located close to the center of Massachusetts' minicomputer industry and excellent opportunities exist for cooperative research and consulting.

The Department has 12 full-time faculty with 200 undergraduates and 50 full-time and 120 part-time graduate students in our M.S. and Ph.D. programs. Department equipment includes 4 VAX 750's, an AT&T 3B-15, 3 Sun Workstations, and over 80 PCs (both UNIX and MS-DOS based). Much of this equipment is networked via two ethernet cables and is also connected to other campus facilities, which include a DEC 2060 and numerous VAXen. The Institute is committed to a new Information Science building and full campus networking facilities in the near future.

Located only 45 miles from Boston, Worcester is a small city of 180,000 which has recently undergone a renaissance. It has eight colleges and universities and a rich variety of cultural activities.

Please send a resume to Karen Lemone, Acting Department Head, Department of Computer Science, WPI, Worcester, MA 01609.

WPI is an Equal Opportunity/Affirmative Action Employer.

NEW JERSEY INSTITUTE OF TECHNOLOGY
Computer and Information Science

Department seeks assistant, associate and full professors for spring/fall 1988. Ph.D. in computer science or closely related field required. Senior level applicants must have proven research and funding record. Positions available in, but not limited to: distributed computing including computer architecture, data communications networking, realtime computing and fault tolerance; software development including artificial intelligence, expert systems, computer graphics; office automation, data management systems, cognitive science, and computational linguistics.

Department offers B.S., B.A., M.S., and Ph.D., in computer science, and Ph.D. in management jointly with Rutgers-Newark. Computing facilities include VAX 8800, VAX 8530, IBM 4361, SUN workstations, Symbolics machines, TI Explorers and graphics systems.

NJIT is the comprehensive technological university of New Jersey with 7700 students enrolled in Newark College of Engineering, the School of Architecture, and College of Sciences and Liberal Arts.

Send resume and names of at least three references to:

Personnel Box C1S
New Jersey Institute of Technology
Newark, NJ 07102 EO/AA employer

NJIT does not discriminate on the basis of sex, race, color, handicap, national or ethnic origin, or age in employment.

UNIVERSITY OF BRIDGEPORT
Computer Engineering

Tenure track positions in Computer Engineering available September 1988. Candidates should have strong commitment to computer engineering education plus strong background in any of the following: computer hardware, digital design, VLSI design, image processing, pattern recognition and computer organization among others. Ph.D. in Computer Engineering preferred but degrees in related areas with extensive computer experience will be considered. The University is located in prestigious Fairfield County, Connecticut within easy driving distance of New York and Boston. The primary goal of the engineering college is instruction, but we are presently developing a larger research emphasis through the Connecticut Technology Institute, initially funded through a five million dollar federal grant. Facilities include VAX 785, DEC 20/60, PC lab, VLSI design lab, AI workstations lab, digital signal processing lab using TI 320 chips connected to PC's, Z80 microcomputer hardware lab, and 8086 microcomputer software lab. Send resume and at least three current references to: Dr. Stephen Grodzinsky, Department of Computer Science and Engineering, University of Bridgeport, Bridgeport, CT 06601.

An Equal Opportunity Employer

CASE INSTITUTE OF TECHNOLOGY
Case Western Reserve University

We invite applications for tenure-track faculty positions at all levels. We are particularly interested in candidates whose research areas include VLSI systems and design automation, applied artificial intelligence, data bases, software design environments, parallel computation, and analysis of algorithms. Applicants will be judged primarily on their ability to strengthen our quest for excellence in research and teaching.

CWRU is a small private university with a total enrollment of about 8400, of which about 5100 are graduate and professional students. The university campus is the hub of the pleasant area known as University Circle, an incorporation with neighboring cultural centers and museums. University Circle is about 10 miles from downtown Cleveland.

Case Institute of Technology, a subunit of CWRU, is among the top ten engineering schools in terms of research funding per faculty member and undergraduate student quality. The department of Computer Engineering and Science has a young faculty of 10 and growing, and a graduate student body of 115 students, 48 of which are currently in the Ph.D. program. Department facilities include a DEC VAX-11/780, Data General MV/10000, numerous desk-top computers (Intel, DEC, Sun and Apollo), several color graphic displays (four high and ten medium resolution), and hard-copy equipment (color ink jet and laser printers, plotters, etc.). Faculty and students participating in the Center for Automation and Intelligent Systems Research have access to the Center's VAX-11/782 and VAXstation 100 display systems. Educational computing is provided by the Computing Center with three DEC-2060's the Case Personal Computer Lab with 48 DEC PRO 350's, four AT&T 3B2's and several AT&T 6300's, and the department's MV/10000.

Applicants should submit their resume and names of three references to: Professor J. Thomas Mortimer, Chairman; Department of Computer Engineering and Science; Case Western Reserve University, Cleveland, Ohio 44106.

An Equal Opportunity Affirmative Action Employer.

FROSTBURG STATE UNIVERSITY COMPUTER SCIENCE FACULTY POSITIONS

The Department of Computer Science is seeking to staff two newly-created tenure-track positions beginning in the Fall, 1988. Master's degree in Computer Science, Information Science, or a related field required. Ph.D. preferred. The candidate should have a desire and the ability to teach a wide range of undergraduate computer courses including some of, but not limited to, the following areas: programming languages, operating systems, architecture, computing theory, graphics, compiler construction, robotics, database, telecommunications, data structures, artificial intelligence and expert systems, as well as introductory level courses. Rank and salary dependent on qualifications and experience. Salary range \$20,000-\$40,000. Send a letter of application, curriculum vita, transcripts and at least three letters of recommendation by April 15, 1988, to: Mr. C. Douglas Schmidt, Director of Personnel Services, Frostburg State University, Frostburg, MD 21532. Women and minorities are encouraged to apply. Frostburg State University is an Affirmative Action/Equal Opportunity Employer.

CLEMSON UNIVERSITY

ELECTRICAL and COMPUTER ENGINEERING at CLEMSON UNIVERSITY is seeking applicants for tenure-track positions in Computer Engineering program offering BS, MS and PhD degrees. All areas of computer engineering are of interest, with special emphasis on distributed system software and hardware, computer networking and computer architecture. Tenure-Track Positions in Electrical Engineering degree program also available, with areas of specialization including microelectronics, reliability, robotics, communications, signal processing, electromagnetics and power. Visiting Positions may also be available. Applicants for Tenure Track positions must have a PhD and a strong interest in teaching and research. Research Associate positions (BS, MS) are also available. Send resumes to Dr. A. Wayne Bennett, E&CE Department, Clemson University, Clemson, S.C. 29634-0915. Clemson is an Equal Opportunity/Affirmative Action Employer.

THE PENNSYLVANIA STATE UNIVERSITY Computer Engineering

Applications are invited for tenure-track and visiting faculty positions at all levels. Candidates from all areas of computer engineering (hardware and software) will be considered. The Computer Engineering Program at The Pennsylvania State University is within the Department of Electrical Engineering which has over 50 faculty members, and approximately 1500 undergraduate majors, 170 graduate students. Candidates should have a Ph.D. in Electrical/Computer Engineering or related areas. There are 13 faculty members within the Computer Engineering Program. Excellent instruction and research computing facilities are available within the Department, College and at the University Computing Center. Please send your letter of application, resume, or inquiries, together with three references to: T. Feng, Computer Engineering Program, Department of Electrical Engineering 129 Electrical Engineering East, Box ES, The Pennsylvania State University, University Park, PA 16802. Deadline for applications is June 30, 1988 or until suitable qualified candidates are selected. "An Equal Opportunity/Affirmative Action Employer."

Education Program Staff Opening: Senior Computer Scientist

The Software Engineering Institute (SEI) is a federally funded research and development center operated by Carnegie Mellon University under contract to the Department of Defense. The SEI's objective is to provide leadership in software engineering and in the transition of new software engineering technology into practice. The SEI Education Program has a staff opening for a Senior Computer Scientist to support the Graduate Curriculum Project, Undergraduate Software Engineering Education Project, and Academic Affiliates Program.

General Qualifications

Candidates for this position should have an understanding of education and the university environment we are trying to serve, as well as an understanding of computer science and software engineering. University teaching experience is required. Candidates must have a commitment to high quality innovative education so that they, along with other senior technical staff, may participate in planning the overall SEI educational strategy.

Responsibilities

This staff member will have responsibility for coordinating the various activities through which the Education Program interacts with the educational community. These activities include:

- Promoting the Use of SEI Educational Materials
- The Education Program produces a variety of educational materials, including curriculum modules, module support materials, books and monographs, and software tools. This staff member will manage the processes by which these materials are disseminated, including tracking their use and soliciting cooperation from educators in the creation of new materials. In particular, this will include managing the collection, development, and dissemination of classroom materials for all levels of software engineering education.
- Coordinating Academic Affiliate Activities This staff member will develop and execute plans for activities with the Academic Affiliates of the SEI. This includes soliciting, reviewing, and summarizing the annual reports from affiliates, assisting affiliates with curriculum development and insertion efforts, and promoting summer and sabbatical visits to the SEI by affiliated faculty.
- Organizing Education Program Conferences and Workshops Currently the Education Program conducts two Faculty Development Workshops and the SEI Conference on Software Engineering Education annually. This staff member will have primary responsibility for planning these events, evaluating their effectiveness, and improving them.

Further Information

Contact:
Dr. Norman E. Gibbs, Director of Education
Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213
(412) 268-7703
ArpaNet: gibbs@SEI.CMU.EDU

The Software Engineering Institute is sponsored by the Department of Defense. Carnegie Mellon University is an equal opportunity/affirmative action employer. U.S. citizenship or resident alien status is required.

RENSSELAER POLYTECHNIC INSTITUTE Faculty Positions Department of Electrical, Computer, and Systems Engineering

Applications are invited for tenure-track faculty positions at all levels. Areas of interest include but are not limited to: computer engineering, architecture and parallel processing, hardware design, and performance evaluation, computer networks, VLSI design, optical computing and communications, knowledge-based systems and artificial intelligence. The ECSE Department is the largest academic unit at RPI and has a rich tradition of research and education. The department is seeking to add faculty who bring an innovative approach to research and teaching. Active programs in computer engineering, solid-state electronics and integrated circuit design, control systems, robotics and automation, information and decision systems, communications and signal processing, electronics and circuits, and fusion plasma systems contribute to a dynamic research environment. In addition to the extensive research facilities of the department, there are opportunities to initiate or participate in interdisciplinary research programs in one of the major research centers of the School of Engineering, including the Center for Integrated Electronics, Center for Interactive Computer Graphics, and Center for Manufacturing Productivity and Technology Transfer. New faculty are eligible for special arrangements including summer support, equipment, graduate student support, and reduced teaching load in order to encourage growth of their research programs. Applications or requests for more information should be directed to:

Dr. Arthur C. Sanderson
Chairman, Department of Electrical,
Computer, and Systems Engineering
Rensselaer Polytechnic Institute
Troy, NY 12180-3590

RPI is an affirmative action/equal opportunity employer.

UNIVERSITY OF ALBERTA Department of Computing Science

Applications are invited for two tenure-track positions at the Assistant/Associate Professor level. Responsibilities include research as well as teaching at the graduate and undergraduate levels. Strong candidates from all research areas will be considered, but areas of special interests include database systems, VLSI/computer architecture, operating systems, numerical analysis and computer graphics.

The Department consists of 36 academic and 28 support staff. Current hardware support includes an Amdahl 5870, a network of four VAX 11/780's and about thirty Sun Workstations, and well-equipped microcomputer and workstation laboratories for graphics, VLSI, and AI research. Access to a Cyber 205 is available.

Salary range \$32,756 to \$57,236 and is commensurate with qualifications and experience. Send curriculum vitae and the names of three references, and up to three reprints or copies of important publications. New Ph.D.'s should also include a copy of their transcript. Apply to:

Dr. Lee J. White, Chairman
Department of Computing Science
University of Alberta
Edmonton, Alberta, Canada
T6G 2H1

Applications will be accepted until June 30, 1988.

The University of Alberta is committed to the principle of equity in employment

UNIVERSITY OF CINCINNATI
Department of Electrical and
Computer Engineering

The Department invites applications for several new tenured or tenure-track faculty positions at all ranks. The Department has recently filled eight new faculty positions and is engaged in an exciting period of further growth. The primary areas of interest are artificial intelligence; neural networks; parallel processing; VLSI system design and testing; computer architecture; software engineering, languages, compilers and data structures; operating systems; computer graphics; fault tolerant computing; micro-processors; computer networks; database systems and all other areas of computer engineering and computer science. Candidates for senior positions are expected to have outstanding records of achievement with vigorous sponsored research program and ability to lead interdisciplinary research centers. All candidates should have a strong commitment to excellence in teaching. Teaching loads are conducive to the establishment of high quality research programs. Earned doctorate is required. Degrees in Computer Engineering or Computer Science are of special interest. Salary and benefits are extremely attractive. We also seek an Associate/Assistant Department Head who will play a leadership role in nurturing the newly established computer engineering degrees and research programs. The Department offers BS, MS, and PhD degree programs in both Electrical and Computer Engineering, with 27 full time faculty, 105 full time graduate students, 425 undergraduate students, and externally funded research of \$1.5M annually. Applicants should send their resume to Professor Vik J. Kapoor, Head, Electrical and Computer Engineering, University of Cincinnati, Cincinnati, Ohio 45221-0030. Positions will be available starting July 1, 1988. Applications will be considered until the positions are filled. The University of Cincinnati is an Affirmative Action/Equal Opportunity employer and encourages and welcomes applications from women and minorities.

UNIVERSITY OF ROCHESTER
Electrical Engineering

We expect to fill several faculty positions in computer engineering or related areas. Currently, we have faculty groups active in VLSI design and testing, robotics, image/signal processing, computer system design automation, and networking. Those appointed would carry out research in computer engineering and teach related material to graduate and undergraduate students. Applicants should have an established research record or, for very recent graduates, a commitment to establish such a record. These positions are expected to be filled at the Assistant Professor level. However, in exceptional cases a higher level of appointment may be forthcoming. Salary commensurate with experience. These are tenure track positions and candidates are expected to hold or to receive soon their Doctorate in Electrical Engineering or Computer Science or a closely related field. We are particularly interested in women and minority candidates. The University of Rochester provides a receptive environment with excellent students and established research. Applicants should send a full resume, a statement of research plans, and copies of relevant publications to: Professor Sidney Shapiro, Chair, Dept. of Electrical Engineering, University of Rochester, Rochester, NY 14627. The University of Rochester is an Equal Opportunity Employer (M/F).

FLORIDA INTERNATIONAL UNIVERSITY
The State University of Florida at Miami
Director—School of Computer Science

Florida International University invites applications and nominations for the position of Director of School of Computer Science. We are seeking an individual with excellent leadership skills, a distinguished record of research, administrative and teaching experience suitable for appointment at the rank of Professor. An earned Doctorate in Computer Science or related field is required. Salary will be commensurate with qualifications.

The Computer Science program was initiated in 1972 and elevated to the status of a School within the College of Arts & Sciences in the Fall of 1987. Currently there are 20 full-time faculty members performing research in many areas of contemporary Computer Science including distributed processing, database management, software engineering, discrete event simulation, computer architecture, analysis of algorithms, logic of computer programs, computer graphics, AI and expert systems. The student population in the school includes approximately 600 undergraduates, 30 Master's students and 10 doctoral students.

The School of Computer Science is expected to play a leadership role in the University's drive to move into the first rank of research universities while maintaining the quality of its excellent undergraduate program. It enjoys strong support from the University administration and will move into a new building in 1989.

University computing is mainly done on a VAX 8800 system. The resources of the school include a VAX 750, a couple of microvaxes, 4 Transputer workstations, Symbolics 3610, a Silicon Graphics IRIS 2400 Graphics system and numerous personal computers, all connected via ethernet. A \$500,000 grant from the State has been instrumental in formulation of plans to modify the computer laboratory in the school. Current promised support calls for the expenditure of \$200,000 towards the purchase of 11 Transputers—some with disks and one with graphics card, about a dozen Sun workstations and a Sun-4 file server. Substantial support for additional equipment has been committed by the university.

FIU is the largest public university in South Florida. It has more than 16,500 students and 600 full-time faculty members. It offers 167 academic programs and courses at the Bachelor's, Master's and doctoral degree levels.

Applications with names of 5 referees and Nominations should be sent to:

Prof. Jai Navlakha
Chairperson—Search and Screen Committee for DCS

School of Computer Science
University Park Campus
Miami, Florida 33199.
(305) 554-2026.

BITNET: NAVLAKHA@SERVAX
FIU is an Equal Opportunity/Equal Access/Affirmative Action Employer.

INTELLIGENCE NEWSLETTER
NEURAL NETWORK NEWS

The INTELLIGENCE newsletter has been first with all the news in neural networks and neuro-computing since 1984. News on scientific breakthroughs, government spending, new companies and new products, important meetings, venture capital. Published monthly; \$295/year. Ask about our free book offer.

INTELLIGENCE
P.O. Box 20008
NY NY 10025-1510
CALL 800-NEURALS/212-222-1123

UNIVERSITY OF DELAWARE
Department of Computer and
Information Sciences

Are you interested in joining the computer science faculty of a growing, dynamic department in an attractive university town within easy traveling distance to New York, Philadelphia, Baltimore, and Washington? The University of Delaware, centrally located on the East Coast, is recruiting for possible openings for tenure-track and visiting faculty positions in the Department of Computer and Information Sciences beginning September 1, 1988. Strong applicants in all areas of computer science are encouraged to apply. Special interest exists for candidates with research expertise in symbolic mathematical computation, parallel processing, artificial intelligence, networking, graphics, programming languages and software engineering.

A Ph.D. degree or its equivalent, and excellence in research and teaching are required. Salary and rank will be commensurate with the candidate's qualifications and experience.

The Computer and Information Sciences Department offers bachelor, master, and doctoral degrees. Resources devoted to academic use in the University Computing Center include: an IBM 3081D, an CDC Cyber 174, a Vax 8600 and Pyramid 98xe both running Unix, and more than 75 microcomputers (IBM PC-XT's, AT's and Macintosh's).

The Department research facilities include various workstations (Symbolics Lisp machines, Micro-Vax II, SUN-3's, and IBM-AT's) and facilities in a joint research lab shared with the Department of Electrical Engineering. The latter includes a VAX-8500, three VAX 780's and various other smaller machines. The equipment is connected to the ARPAnet, CSNet, and to BITNET.

Candidates should send a curriculum vitae and the names of three references to Professor Claudio Gutierrez, Department of Computer and Information Sciences, University of Delaware, Newark, DE 19716. Positions are open until filled. The University of Delaware is an equal opportunity, affirmative action employer. Applications from members of minority groups and women are encouraged.

NAVAL POSTGRADUATE SCHOOL,
Monterey, CA

Position Announcement in Computer Science

The Department of Computer Science has immediate openings for faculty positions at all levels. Our primary interests are in the areas of operating systems, programming languages, and algorithms. Our secondary interests are in the areas of processing of visual data, real-time systems, and software engineering. An applicant should have a Ph.D. in Computer Science or a related field and have a strong interest in both graduate teaching and research. Senior applicants must have distinguished research records. Appointments can begin at any time during the year.

The Department offers MS and Ph.D. degrees in Computer Science supported by well-equipped instructional/research facilities and full-time technical staff. The faculty normally teach for two quarters and conduct full-time research during the other two quarters.

Please send a detailed resume and three letters of reference to: Search Committee, Computer Science Department (Code 52), Naval Postgraduate School, Monterey, CA 93943, tel. #408) 646-2449. NPS IS AN EQUAL OPPORTUNITY/AFFIRMATIVE ACTION EMPLOYER.

UNIVERSITY OF WASHINGTON

The Department of Computer Science may have openings for tenure-track faculty appointments starting in the 1988-89 academic year. We are particularly interested in applicants with research strengths in artificial intelligence, databases, and programming languages and compilers. However, applications from outstanding individuals in other areas might also be considered. A moderate teaching load allows time for quality research and close involvement with students. We expect applicants to have a strong commitment to both research and teaching, and an outstanding record of research for their level. Any appointment should bring significant new research strength to the department. The department may also have several visiting positions that would require both teaching and research. It may be possible to hold these for portions of the 1988-89 academic year. Interested applicants should send a letter of application, a resume, and the names of four references to Paul Young, Chairman, Department of Computer Science FR-35, University of Washington, Seattle, Washington 98195. The University of Washington is an Affirmative Action/Equal Opportunity Employer. The Ph.D. is required for these positions.

MISSISSIPPI STATE UNIVERSITY

Mississippi State University invites applications for tenure-track computer science faculty positions beginning August 1988 or January 1989. With a newly approved Ph.D. program, applications for both senior and junior positions will be considered. A doctorate in computer science or computer engineering or a record of established research is desired. All applications, however, will be considered. Interested individuals should forward a vita and names of at least three references to: B. D. Carter, Head, Department of Computer Science, Drawer CS, Mississippi State, MS 39762. MSU is an Affirmative Action/Equal Opportunity Employer.

UNIVERSITY OF NORTH DAKOTA

Applications are invited for an anticipated tenure-track faculty position in the Computer Science department at Assistant, Associate, or Professor level. Starting date is August 16, 1988. Candidates must hold a Ph.D. in Computer Science or related field. Computer Science is one of the departments in the Center for Aerospace Sciences, a degree granting college of the University of North Dakota. The department offers an undergraduate major with 250 enrolled and Master's program with 15. A Ph.D. program is anticipated within five years. The department is the principal user of a VAX 11/785 and PDP 11/44, has access to two IBM 3090's, Concurrent 3260 and 3280 and a VAX 780. Computer Science offices and laboratories are in the newest academic building on campus. Salary is open and competitive. Benefits include TIAA-CREF and health and life insurance. A group disability insurance is also available. The candidate must be either a U.S. citizen or eligible to work in the United States. Deadline to apply is April 1, 1988, or until the position is filled. Send resume and three letters of reference to: Dr. Lonny Winrich, Department of Computer Science, Box 8181 University Station, Grand Forks, ND 58202. UND is an equal opportunity affirmative action institution.

CURTIN UNIVERSITY OF TECHNOLOGY

The School of Computing and Quantitative Studies provides an applied computing orientation to students in business information systems. The programmes offered range from Bachelor of Business (Information Processing or Information Systems) to PhD in Information Systems.

The School has a growing international reputation in a number of aspects of Information Systems namely Information Systems Planning and Strategy Formulation, System Development Methodologies, Decision Support Systems and End-user Computing.

It is looking to expand its activity in these and related fields such as Software Engineering, Systems and Software Quality Assurance, System Design Approaches, Automated Support Tools, Executive Information Systems and Commercial Exploitation of Expert Systems. Two Tenured Senior Lecturers (Ref 1089) are required to take a leading role developing research and educational programmes in one or more of the above fields. In addition to teaching, appointees will be required to engage in research/scholarship; provide leadership/guidance to both staff and postgraduate students and monitor the relevance of the School's undergraduate/postgraduate offerings in the light of current trends.

A PhD or Master's Degree is required together with a record of applied research and academic leadership and relevant professional experience, preferably at a senior level.

Four Lecturers (two tenured, two limited term) (Ref-1090) are also required to teach and participate in research and educational programmes in the following areas:

Program Design: Knowledge of COBOL and the use of System Software such as screen handling, transaction processing and database manipulation and experience in an IBM/MVS environment with practical experience in on-line and batch data processing would be desirable.

Systems Development: A good understanding of SD methods including information requirements determination, knowledge elicitation, data analysis (normalization and entity modelling), systems design, specification and testing is required. Experience of project management in a business setting would be an advantage.

Systems Software (PCs): A broad-based knowledge of operating systems including hardware interfacing, a strong end-user orientation and a practical appreciation of all hardware and software aspects of personal computers is required. Additionally, experience in a mainframe environment would be a definite advantage.

At the Lecturer level a postgraduate qualification is preferred although candidates with an undergraduate qualification and extensive relevant professional experience will also be considered. **Salary range:** (\$Aust) Senior Lecturer A\$37,903 -A\$44,090; Lecturer A\$28,381 -A\$37,122. Limited Term Appointments are available initially for one to three years and to a potential maximum of five years.

Applications: Details including names, addresses and telephone numbers of three referees should be submitted not later than 8 April 1988 to the Appointments Officer, Curtin University of Technology, GPO Box U1987, Perth, Western Australia, 6001. Further information may be obtained by Telex (AA 92983) or Fax (619) 458 4661 quoting "Appointments", position reference number and your return airmail address or by telephoning in business hours Australia (619) 350 7064.

OHIO STATE UNIVERSITY Faculty Positions in Human Factors Engineering

Applications and nominations are invited for tenure track positions in the Department of Industrial and Systems Engineering at the Ohio State University. A Ph.D. degree is required. Preference will be given to applicants with strong background and research interests in cognitive engineering (especially interests in artificial intelligence, human-computer interactions and/or human problem solving) or biomechanics. Applications from women and minorities are encouraged. Individuals will be expected to teach and develop undergraduate and graduate courses in human factors and develop research programs which compliment and support existing programs.

Nominations and applications consisting of a resume and the names and addresses of three references should be sent to:

Human Factors Search Committee
Department of Industrial and Systems Engineering
The Ohio State University
1971 Neil Ave.
Columbus, Ohio 43210

UNIVERSITY OF TENNESSEE SPACE INSTITUTE Graduate Research Assistants in Computer Science

Applications are invited for graduate research assistants in computer science, particularly in artificial intelligence. UTSI awards M.S. and Ph.D. degrees in computer science with emphasis on applied artificial intelligence, expert systems and robotics.

UTSI offers total financial assistance packages including tuition, fees, and monthly stipends which average \$11,600 to \$12,600 for an academic 9 month appointment depending upon the degree program. In addition, summer appointments may be available. To receive an application or for further information, please contact: Dr. Charles Lea, Director, Admissions and Student Services, The University of Tennessee Space Institute, Tullahoma, TN 37388 (phone: (615) 455-0631, ext. 296).

UTSI is an AA/EEO employer

OREGON STATE UNIVERSITY Department of Computer Science

The Department of Computer Science invites qualified applicants for two graduate fellowships in Computer Science at Oregon State University. These will begin with the Fall Quarter of 1988 and will carry a stipend of \$10,000 per year. Recipients will be selected on a competitive basis, with undergraduate performance, scores on the Graduate Record Examination, and references as the primary source of information. We expect that recipients will enroll in the Ph.D. program in Computer Science at OSU and will devote their full time toward the pursuit of that degree. For further information and for application materials, please contact

Walter G. Rudd, Chairman
Department of Computer Science
Oregon State University
Corvallis, Oregon 97331
503-754-3273

Oregon State University is an Affirmative Action/Equal Opportunities employer and complies with Section 504 of the Rehabilitation Act of 1973.

International Symposium on Databases in Parallel and Distributed Systems

December 5-7, 1988
Austin, Texas



Symposium General Chair
Joseph E. Urban
University of Miami

Co-Program Chairs
Sushil Jajodia
NSF
Won Kim
MCC
Avi Silberschatz
UT-Austin

Program Committee
Rakesh Agrawal, *AT&T Bell Labs*
Francois Bancilhon, *INRIA*
John Carlis, *U. of Minnesota*
Doug DeGroot, *TI*
C. Ellis, *Duke U.*
Shinya Fushimi, *Japan*
H. Garcia-Molina, *Princeton U.*
Theo Haerder, *Germany*
Yahiko Kambayashi, *Japan*
Gerald Karam, *Carleton U.*
Michael Kifer, *SUNY-Stony Brook*
Roger King, *U. of Colorado*
Hank Korth, *UT-Austin*
Ravi Krishnamurthy, *MCC*
Duncan Lawrie, *U. of Illinois*
Edward T. Lee, *U. of Miami*
Eliot Moss, *U. of Mass.*
Anil Nigam, *IBM Yorktown Heights*
N. Roussopoulos, *U of MD*
Sunil Sarin, *CCA*
Y. Sagiv, *Hebrew U.*
Jlm Smlth, *ONR*
Ralph F. Wachter, *ONR*
Ouri Wolfson, *Technion*
Clement Yu, *UI-Chicago*
Stan Zdonik, *Brown U.*

Local Arrangement
Hong-Tai Chou, *MCC*

Publicity
Ahmed K. Elmagarmid, *Penn State*

Finance Chairman
Edward T. Lee, *U. of Miami*

Sponsored by:

IEEE Computer Society Technical Committee on Data Engineering & ACM
Special Interest Group on Computer Architecture (Approval Pending)

In Cooperation With:

IEEE Computer Society Technical Committee on Distributed Processing

Symposium Objectives

The objective of this symposium is to provide a forum for database researchers and practitioners to increase their awareness of the impacts on data models and database system architecture of parallel and distributed systems and new programming paradigms designed for parallelism. A number of general purpose parallel computers are now commercially available, and to better exploit their capabilities, a number of programming languages are currently being designed based on the logic, functional, and/or object-oriented paradigm. Further, research into homogeneous distributed databases has matured and resulted in a number of recently announced commercial distributed database systems. However, there are still major open research issues in heterogeneous distributed databases; the impacts of the new programming paradigms on data model and database system architecture are not well understood; and considerable research remains to be done to exploit the capabilities of parallel computing systems for database applications.

We invite authors to submit original technical papers describing recent and novel research or engineering developments in all areas relevant to the theme of this symposium. Topics include, but are not limited to,

- Parallelism in data-intensive applications, both traditional (such as Transaction Processing) and non-traditional (such as Knowledge-Based)
- Parallel computer architecture for database applications
- Concurrent programming languages
- Database issues in integrated database technology with the logic, functional, or object oriented paradigm
- Performance, consistency, and architectural aspects of distributed databases

The length of each paper should be limited to 25 double-spaced typed pages (or about 5000 words). Four copies of completed papers should be sent before May 1, 1988 to:

Won Kim, MCC, 3500 West Balcones Center Dr., Austin, TX 78759
(512) 338-3439, kim@mcc.com

Papers due: May 1, 1988
Notification of Acceptance: July 15, 1988
Camera-ready copy due: August 30, 1988

BOOK REVIEWS

Editor: Wiley McKinzie, School of Computer Science and Technology, Rochester Institute of Technology, Rochester, NY 14623; Compmail, w.mckinzie; CSnet, wrm@rit

Nations at Risk: The Impact of the Computer Revolution

Edward Yourdon (Yourdon Press, New York, 1986, 616 pp., \$18.95)

Most readers are aware that the number of books dealing with the impact of computer technology on society has grown exponentially over the last several years. Having received graduate degrees in both computer science and sociology, I have been keenly interested in this field and have generally been disappointed with the offerings. Only a few books—for example, *Social Issues in Computing* (Gottlieb and Borodin, Academic Press, 1973)—have presented the public with a serious treatment of the subject. It was, therefore, with both anticipation and apprehension that I began reading *Nations at Risk*. Yourdon's book proved to be a delightful addition to the short list of noteworthy books assessing the technological impact of the computer revolution.

Yourdon addresses his remarks to the responsible citizen rather than the computer scientist. This is a book meant not only to inform readers but to spur them to take an active role in determining the future impact of computer technology. It would thus make an excellent text for a computer literacy or similar course. In addition, the book is written in an informal, personal style that makes it accessible and enjoyable to the general reader; Yourdon explains the technical terminology at a level the layperson can understand. Indeed, Yourdon's ability to integrate his personal experiences into a discussion of complex technological issues increases the book's value.

This book is not, however, an "ordinary" text on the impact of computer technology. As its title, inspired by the 1983 report of the Presidential Commission on Excellence in Education, suggests, Yourdon believes that the United States and the world community are equally at risk from a technology that is developing faster than our ability to adapt to it. Unless we attempt to understand and mold this technology rather than simply respond to it, Yourdon thinks we will truly be *Nations at Risk*.

Thus, Yourdon seeks to provide any person living in the computer age with the necessary information to understand current technological developments and to participate actively in decisions regarding them.

Yourdon divides his discussion of these developments into seven broad sections, subdivided into 39 chapters. Each section discusses a major institution affected by computer technology—government, society, business—or a major area of the technology itself—the computer industry, computer software, or computer hardware. A final section is devoted to wider issues that affect not only computer technology but humanity in general, for instance, nuclear war, pollution, etc. Each of the topics is presented separately. This means the reader can choose to read at random about topics of personal interest. Unfortunately, it also means the book is repetitious. For example, Yourdon states at least five times that "between 50% and 80% of the software work done in most organizations is associated with revision, modification, conversion," etc., of previously existing programs. Such repetition detracts somewhat from the overall quality of the book, but it is understandable, given the author's orientation.

Perhaps a greater detraction is the author's repetition of epithets. For example, while I might agree with Yourdon's portrayal of the Basic programming language as "(ugh!) BASIC," the use of this epithet whenever Basic is discussed becomes tiresome. Similar irrelevant editorial comments on the efficiency of the post office and the general appearance of programmers occur in the text. However, these are only minor irritants in an otherwise excellent text.

While it is impossible to summarize adequately a book of this breadth in a brief review, I would like to examine two of the sections, Computers and Society and Computer Hardware, to provide a flavor of Yourdon's cover-

age. The section on computers and society introduces a broad range of topics, including robotics, computerized home control, computers in the banking industry, the growth of a new cottage industry among computer professionals, video games, and computers and children. For each topic, Yourdon describes the present situation and indicates the short- (five years hence) and probable long-term trends. In addition, he suggests courses of action depending on the reader's position within society, for example, what the responsible parent, citizen, and/or computer professional should do. He also includes a list of references for future readings.

The chapter on computers and children illustrates the richness of Yourdon's discourse. Here he outlines current computer use within the school system, for instance, simulation, programming, drill and practice, and game playing, including the benefits and detriments from each type of use. In addition, he discusses problems such as gender bias and economic inequality in computer use. Advice is given to parents and teachers. For example, Yourdon recommends drawing programs for younger children (third grade or below) and the use of Logo for older children. It is interesting that the drawing program he recommended for my computer listed for \$395. While I hope I am a responsible parent, this seems a rather high price to pay for the development of my five-year-old daughter's, possibly nonexistent, artistic talents. I suspect that many other parents may react in a similar fashion.

Yourdon's examination of hardware technology is equally comprehensive. For the novice, he provides a highly readable synopsis of major computer components, that is, CPU, memory, printers, and disks. He analyzes the current hardware growth rate and makes predictions concerning future growth. In addition, he expertly summarizes new technological developments, including parallel processing, gallium arsenide chips, and optical and organic

computers. While his analysis would probably not present any new information to a computer scientist or an engineer, it does provide the average person with a superb sketch of current developments.

In conclusion, *Nations at Risk* is excellent reading. It is the best overall portrayal of computer technology that I have read. It is current, comprehensive, and personal. It will be enjoyed by citizen, student, and computer professional. I would highly recommend it as a text in a computer literacy course, particularly if a softbound version is published.

Susan Anderson-Freed
Illinois Wesleyan University

Software Engineering: Planning for Change

David Alex Lamb (Prentice-Hall,
Englewood Cliffs, N.J., 1988, 298
pp., \$47.00)

This book provides a useful overview of important topics in software engineering and would be an appropriate textbook for computer science majors at designated undergraduate and graduate levels. Lamb emphasizes the consistent examination of issues related to programming in the large, and the text would be well received in a senior-level projects course or similar advanced group.

I enjoyed the author's realistic approach to software development and maintenance. He tells it like it is—just the way it happens in the messy, confusing, and bureaucratic world confronting software engineers. I often find that related textbooks are oriented more toward the cut-and-dried aspects of software engineering, using a typical teaching-style approach to focus on small, individual efforts for developing software. Fortunately, Lamb has taken a different approach.

The book's 20 chapters are divided among four sections—Overview, Software Lifetime, Specifications and Verification, and Other Topics—and there are six appendices. Each chapter does what a good textbook should: starts with a catchy introduction, lays out the topics in a straightforward manner, finishes with a chapter wrap-up, and provides exercises, project suggestions, and further readings. Though I was pleased to see all of the essential ingredients, in some cases the book

seemed thin on the readings, exercises, or project ideas. An instructor could get by with the minimum provided but would more than likely have to supplement these classroom-oriented items.

Chapter 1 introduces the field of software engineering with a look at the question of “programming versus software engineering” as both title and occupation. A useful analogy suggests that a programmer is a technician, a software engineer is an electrical engineer, and a computer scientist is a physicist. The end of the chapter offers several good recommendations for readings (e.g., Brooks, Parnas, Glass), but other “greats” have been left out. (It turns out that the few authors named here are repeatedly cited throughout the book.)

What can a life cycle concept offer software engineers? Chapter 2 looks into this debate. Lamb takes the position that the famous waterfall is good enough to help out, but he knows that we know (and he lets the student know) that all is not well with the concept. I liked his discussion of software tools, particularly his reminder that tools can change the behavior of developers, and the behavior of developers can change tools. Recommended reading includes a general reference to the ACM issues on Software Engineering Notes that Lamb should have made a bit more specific.

Next comes a surprise chapter on technical writing. I often urge my students to write—not just programs, mind you, but prose and documentation—yet many books skip the issue of technical writing altogether. (Of course, students will argue that this is a computer class, not an English class.) Some books push the writing material all the way to the end; here we get it right up front. The exercises given at the end of the chapter are especially relevant and should be assigned.

Chapter 4, on requirements analysis, uses a few finite-state-machine examples to illustrate the major points. The chapter is sufficient but not a knockout. Exercises given at the end are pretty good, and later chapters build on these problems to create progressively larger and larger assignments (problems included deal with automated calendaring, library-book checkout systems, computer board games, home budgeting packages, and programmable calculators).

In Chapter 5, dealing with preliminary design, the author introduces some mathematical notation to express design features. For the moment, the math doesn't help much, but in later chapters the notation becomes more useful. One of my favorite passages in this chapter

concerns information hiding and data abstraction: “Information hiding focuses on what to hide; abstracting focuses on what to reveal” (p. 48). I also liked the use of Pascal-ish-looking code to show different implementations of stacks (one time using a list, another time an array).

Chapter 6 is on module interfaces and includes the first extensive use of code in the book. The code is PDL-like, Pascal-like, Ada-like and is shown in very brief fragments (perhaps a half page on the average). Obviously, students reading the book should already know about code and should be comfortable looking at incomplete fragments and able to catch on to syntax/semantics on-the-fly. I found the code appropriate for the intended audience (though it's always nice to have some fuller examples given).

The next few chapters follow the trend started by Chapter 6. First, Chapter 7 examines module implementation issues such as namespace, exceptions, comments, undefined values, and so on. I was disappointed that no references were given here. Next, Chapter 8 examines testing (logic, overload, timing, performance, etc.). Then Chapter 9 looks at system delivery, but the material isn't as lively as in prior chapters and seems to lose the earlier realism (in addition, there are no references or exercises). Finally, Chapter 10 takes on the evolution of software and examines issues such as maintenance.

Chapter 11 starts a new trend by moving into more mathematical and theory-based views of software engineering. The next several chapters examine (or reexamine) life cycle concerns with a new set of lenses. I believe the material would be a good introduction to the topics for most students, but it falls short of really showing the power that results from the shift in emphasis.

After reading 168 pages, I was ready for a strong show in the fifteenth chapter, entitled “The Workplace.” Somehow Lamb lost his bite, and the chapter is merely traditional. Even less exciting is the next chapter on scheduling (not another look at simple PERT!). Chapter 17 is on configuration management, and though in general it covers the material, I prefer such coverage to be more specific and to identify actual packages available in the marketplace. Chapter 18 is on a topic of increasing visibility and importance: software quality. Again the coverage is adequate but misses in its specificity (and no references are given to even guide a student toward further exploration). Worse still, Chapter 19, on tools, omits

any precise details on specific tools.

The concluding chapter admits to some of the topics the book didn't get around to. The appendixes include an example program (mentioned throughout the textbook) dealing with a small genealogical database. They also include various documentation sets and the source code to the single sample program (written in IBM Pascal/VS). A short glossary, bibliography, and index complete the book.

Overall, I recommend the book as a viable candidate for use in a senior-level undergraduate computer science course. Keep in mind that the students should already be comfortable with programming and interested in expanding their horizons. If the Lamb book is used carefully, students can gain some real-world insight into software engineering along with the theory. Instructors would probably want the course to be projects oriented and would use the book as a departure point for discussion (as opposed to direct lecture material). Weak as it is in some respects, I found the book enjoyable and usable for a classroom setting.

Lance B. Eliot
University of Southern California

IBM Personal System/2: A Business Perspective

James W. Hoskins (John Wiley & Sons, New York, 1987, 242 pp., \$19.95, softcover)

Though not officially so, this is an IBM book through and through. The author is an IBM engineer, the Foreword is by an IBM manager, and the book "was reviewed by more than 50 different IBM engineers." (To round out the picture, this reviewer is a retired IBM engineer!) It's a comprehensive nontechnical review of IBM's recently announced (April 1987) successor to the IBM PC microcomputer line, the PS/2 family. Can we still call them "micros"? The PS/2 Model 80 can have up to 16M bytes of main memory, 230M bytes of hard disk storage, and 1600M bytes of removable-media optical storage.

This book is not a technical reference manual. It is a comprehensive description of the PS/2 Models 50/60/80 from a nonengineering perspective. It meets the needs of high-level-language programmers, planners, prospective buyers, users, and anyone else who

wants to understand PS/2 and OS/2 (Operating System/2) with no more than a glance inside the covers. It will be particularly satisfying to those who have tried to assemble a coherent picture of the new family from the myriad IBM "letters," brochures, and guides—or the reviews, overviews, and summaries in the trade press. It is also a useful primer that will give the future PS/2 user a running start while awaiting delivery. It should be comprehensible to the computer neophyte, but it's really aimed at the experienced nontechnical PC user.

Appropriately, the (non-OS/2-capable) Model 30 is dismissed with a few early paragraphs and a picture as PS/2's legacy from the old PC family. The 30's 8086 processor and 3½-inch disks are noted, but the dramatically improved graphics are not addressed. The Model 25 was announced too late to even be mentioned.

Having an "insider" author enabled the book to be available very early after PS/2's announcement. On the down side, the book preaches only the IBM "party line." It espouses the virtue of open architecture but rarely mentions products other than IBM's, except for briefly listing, in the compatibility appendix, the handful of non-IBM software products tested. This is partly excusable because of the newness of the PS/2 family and the book's early publication date.

A seven-page table of contents provides detailed access to the text, making the book a useful reference source. The index is less useful, actually only a glossary of terminology with pointers to the pertinent text pages.

The book's seven chapters and five appendixes are logically organized. It's well illustrated with photos, sketches, and screen images. Three separate chapters competently treat the system overview, options and peripherals, and communications.

One chapter is primarily a detailed walk-through describing use of the PS/2 with the reference diskette that accompanies each machine. This diskette contains a comprehensive user's tutorial and a handful of setup utilities.

A nine-page chapter on application programs treats the subject quite superficially, except for pointing out the increased potential for powerful business graphics offered by the new VGA (Video Graphics Array) capability.

The final chapter addresses, in a necessarily simplistic way, the choice of the proper PS/2 model to fit the user's needs. It offers separate scenarios for small, medium, and large businesses, with proposed configurations for each.

A 16-page appendix contains IBM's publication of an outside testing lab's performance tests of the various PS/2 models versus the PC XT. The listings of software compatibility published by IBM when PS/2 was announced are reprinted in a 40-page appendix. These listings (mostly for IBM software) cover both PC/DOS 3.3 and OS/2 (in DOS-compatibility mode). The other appendixes—on the Model 50, other PS/2 publications, and peripheral compatibility—have rather trivial content.

Overall, this is a useful book for those who need a not-too-detailed introduction to the mainstream IBM PS/2 family (i.e., Models 50/60/80). It won't give you much insight into the Micro Channel, or details of the 80386 implementation, and it won't present much of a technical challenge. But it does provide authoritative information, easily accessible, in one place.

Charles B. Stott
Charles B. Stott Associates

Handbook of Software and Hardware Interfacing for IBM PCs

Jeffrey P. Royer (Prentice-Hall, Englewood Cliffs, N.J., 1987, 254 pp., \$24.95)

This book gives excellent information on the specifics needed to write good interface programs, as well as information to build rudimentary interface hardware. While its scope is limited strictly to IBM PCs, most of the information can be used for other members of the PC family, as well as most clones. The chapters are split between hardware- and software-interfacing topics; however, one cannot describe software interfacing without first describing the hardware. Consequently, Royer has devoted the first two-thirds of the book to software interfacing, accompanied by descriptions of each specific piece of hardware; the last third relates more to hardware interfacing.

The book is targeted for somewhat experienced systems programmers, and hardware/software engineers. An overview and basic understanding of the IBM PC hardware is needed to fully understand some of the concepts presented. While the IBM PC may seem a bit archaic as the choice of hardware, the information presented provides a good foundation for the continuing line

of PCs, namely the newer class of 80286 and 80386 machines. The book's main purpose is to present an interface guide that the engineer can refer to when generating a specific interface, be it hardware or software. Not only is the book an excellent text for the interface developer, it is a must for anyone wanting to write low-level programs that require any type of input/output. It also does an excellent job of discussing most of the DOS interrupts and functions.

The text itself is derived from a course taught by the author, although it could just as well be used as a reference book for the experienced interface programmer. The text contains very few references to other sources, and when they do appear, they point mostly to the PC and the DOS technical reference manuals. Actually, the work is self-contained and requires very few references. An index is included.

The experienced programmer will find that this book shines in comparison with others dealing with PCs, for example, Peter Norton's *Inside the IBM PC*, (Brady Communications Company, 1986). Royer's text skips the basic and often boring fundamentals, which should already be familiar to readers involved with interfacing. It devotes its space instead to a discussion of the PC's internal workings, even to the point where interfacing is just another type of application.

The excellent layout provides a very good mix of text and white space. Large chapter headings and bold topic titles make the text extremely attractive. Assembly language examples are abundant throughout, as are pictorial descriptions of various topics. Each chapter has a short, one-paragraph summary to reinforce what the reader has learned. The logical division of the book into hardware and software enables a reader to locate pertinent information in a matter of seconds.

Chapters 1-3 contain introductory material, including a broad overview of the IBM PC hardware and a high-level look at the structure of DOS. Chapters 4-11 describe in depth the disk operating system and the basic input-output system. Chapters 12-18 contain information on display adapters, the keyboard, CPU, I/O channel, direct memory access, and hardware interrupts.

On the software side, the book contains a wealth of information pertaining to the inner workings of DOS. Unlike other material on the subject, where just one interrupt is recognized, this text includes all eight DOS interrupts. It is important for the user to know that even though one DOS interrupt handles 99 percent of the traffic flow, the other

interrupts do exist and are used. The author uses three chapters to explain disk structure, describing the file allocation tables as well as the various methods to access and use files. One chapter discusses the differences between .EXE and .COM files and the structures of each. Another chapter describes the DOS executive functions used to manage memory usage and allocation.

Royer devotes one chapter to installable device drivers. While both character and block drivers are mentioned, he focuses on character drivers. In addition to an in-depth explanation of a driver, Royer also gives an assembly language example of a driver.

On the hardware side, Royer gives some insight into programming the 8237A DMA controller chip and the 8259A interrupt controller. Descriptions of the monochrome adapter and Color Graphics Adapter are included. However, a description of the EGA adapter is lacking. The book could be improved by including the EGA adapter and omitting the introduction to logic gates. The operation of the keyboard, keyboard interrupt, and I/O is discussed. An excellent example of a "hot key" to activate a memory-resident program is given. The hardware chapters conclude with information on buses, bus cycles, I/O channels, I/O ports, and timing requirements for interface cards.

Royer has succeeded at presenting the reader with a handbook for interfacing to the IBM PC and has also provided a good source of information for the user wishing to learn the inner workings of DOS.

Jeff S. Ebert
Unisys Corporation

Microcomputer Hardware Design

D.A. Protopapas (Prentice-Hall, Englewood Cliffs, N.J., 1988, 510 pp., \$42.67)

There are a number of ways to present the concepts of computer architecture and design. One approach, often used in architecture classes for undergraduate computer science majors, is to present the computer as a tool the student must understand in order to use properly, but not a tool the student will be building. This approach provides the computer science major with the information needed to master the operations of the tool.

Another approach, more appropriately applied to electrical engineering and computer technology majors, is to present computer architecture and design as a field to be mastered. This approach provides the knowledge and skills required to select and assemble the building blocks needed to create the tool (computer) and make it operational. Such is the goal of this text, and it is well met.

Beginning with the organization of a prototypical computer, the author leads the reader in logical progression through the design process. Basic questions are addressed: Why should we use microcomputers? What kind of components are available? (Obviously just a slice from the evolving microcomputer technology can be presented.) What are the typical design goals? How do we get from here to a full-up system?

Microcomputer design is presented in a building-block fashion, with emphasis on standard von Neumann machines using a general-purpose CPU. The CPU, memory (both RAM and ROM), input/output techniques, and peripherals are discussed in sufficient detail to provide the basic knowledge needed to proceed with the design process, or to examine and understand the concepts and techniques behind a design.

I found Chapter 3 of particular interest. This chapter presents, in detail, the design of several popular 8-, 16-, and 32-bit CPU chips from the Intel 808xx and Motorola 680xx families. It also discusses the typical support chips available for the various families of CPUs. Having all this material presented in a single package and a format that is uniform is very valuable.

To support the material provided in each chapter, the author has also included design examples in each area of interest, and each chapter contains an extensive set of problems that help reinforce and expand the reader's understanding of the material.

In the Preface the author states that the text is "... beyond the introductory level ...," and so it is. While no prior knowledge of microcomputer design is assumed, the reader must have some knowledge of computers, programming, computer organization, and logic design. The text could serve well in a two-semester senior- or one-semester graduate-level electrical engineering course. As a self-study text for working engineers, it could be improved by the inclusion of solutions to selected problems, and an expanded reference list.

Alan E. Gould
Synetics Corporation

NEW LITERATURE

Programming techniques. Seven high-level programmers provide insights into how they develop program code in a 20-page booklet entitled "Editing Techniques of World-Class Programmers," available free from Solution Systems. The information is presented in interview format. Contact Solution Systems, 541 Main St., Suite 410, South Weymouth, MA 02190; (617) 337-6963 or (800) 821-2492.

The Computer Graphics/Desktop Publishing Directory and Buyers Guide is scheduled for publication in April 1988, with updates every six months. The directory is a joint venture between Directory House and The Oryx Press. It will cost \$50.

For more information, contact Directory House, 27292 Calle Arroyo, San Juan Capistrano, CA 92675; (714) 240-0491.

Software legal guide revised. Nolo Press has revised and updated *Legal Care for Your Software* by Daniel Remer and Stephen Elias for the third edition (ISBN 0-87337-037-6, 352 pp., \$29.95). The latest edition incorporates legal developments and changes in the software industry and its trade customs. The book covers ways to protect software and what to do if infringement occurs. Order from Nolo Press, 950 Parker St., Berkeley, CA 94710; (415) 549-1976.

Programming with Windows (ISBN 0-88022-299-9, Order #99, 450 pp., \$22.95) by Tim Farrell provides a guide to creating applications programs with Microsoft's Windows 2.0 or 386. The book builds on the reader's knowledge of the C programming language. A companion software program costs \$39.95 (Order #277). Both are available from Que Corp., 11711 N. College Ave., Carmel, IN 46032; (317) 573-2544.

Superconductivity. The American Physical Society has collected 112 papers on superconductivity published during the first six months of 1987 in the APS journals *Physical Review Letters* and *Physical Review*. The reprint volume *High-Temperature Superconductivity* (ISBN 0-88318-539-3, 400 pp., \$35 non-members, \$25 APS members) is available from the American Physical Society, Publications Liaison Office, 500 Sunnyside Blvd., Woodbury, NY 11797; (516) 349-7800, ext. 604.

New ideas, creative approaches and bold leadership are needed for America's process manufacturing industry to retain its traditional place as world leader. Honeywell and Arizona State University have joined forces to create a dynamic new graduate study program with the goal of developing young leaders able to meet the manufacturing challenges of the global marketplace of the future.

Honeywell Industrial Fellows Program

The Department of Computer Science at Arizona State University invites applications for its 1988 Honeywell Industrial Fellows Program. Sponsored by the Industrial Automation Systems Division of Honeywell, Inc., located in Phoenix, this program offers an excellent opportunity for up to two outstanding engineers to earn a masters degree in computer science or computer engineering from one of the nation's best engineering colleges, and, at the same time, gain valuable work experience at one of America's top developers of process manufacturing systems.

Terms of Awards: This program will combine a two year master of science in computer science degree program, covering the 1988-89 and 1989-90 academic school years, with part-time employment (20 hours per week) at Honeywell. Fellows will be employed by Arizona State University as graduate research assistants. A salary of \$11,000, plus a full tuition waiver, health insurance, and a book allowance will be provided for each academic school year, and Fellows will have the option of 1988 summer employment with Honeywell.

Selection criteria: The ideal applicant will have a bachelors degree from an ABET-accredited institution in computer science or computer or electrical engineering. Relevant work experience is a plus, as are excellent grades. U.S. citizenship is required. Arizona State University vigorously pursues affirmative action in its employment, activity and programs.

Applications are due April 30.

For more information contact:
The Honeywell Industrial Fellows Program
Department of Computer Science
College of Engineering and Applied Sciences
Arizona State University
Tempe, AZ 85287-5406
(602) 965-3190

Computer Science

Arizona State University

ADVERTISER INDEX

Advanced Micro Devices	103
Aerospace Computer Security Applications Conference—4th	90
Arizona State University	159
CACI	1
CASE '88	Cover III
Classified Advertising	149-153
Computer Architecture Symposium—15th	104
Computer Networking Symposium	118
Computer Society Membership	133
Computer Society Ombudsman	49
Contel Technology Center	4-5
DAC—25th	127
Data Engineering Conference—5th	139
Databases in Parallel & Distributed Systems Symposium	154
Engineering of Computer-Based Medical Systems Symposium	141
Frontiers '88	121
ICC '88	7
ICCAD '88	89
Infocom '89	10
Information Research Associates	Cover II
Institute for Defense Analyses	75
MCC	48
Meta Software	23, 64
MIT Press	21
MITRE Corporation	40
NeuralWare Inc.	117
SAIC	Cover IV
Security & Privacy Symposium '88	50-51
Sigmatrics '88	76
Visible Systems Corporation	24

FOR DISPLAY ADVERTISING INFORMATION CONTACT

Northern California and Pacific Northwest: Don Farris Co., 161 W. 25th Ave., Ste. 102B, San Mateo, CA 94403; (415) 349-2222.
Jack Vance, P.O. Box 3205, Saratoga, CA 95070; (408) 741-0354.
Southern California and Mountain States: Richard C. Faust Co., 24050 Madison St., Ste. 100, Torrance, CA 90505; (213) 373-9604.
Southwest: The House Co., 3000 Wesleyan, Ste. 345, Houston, TX 77027; (713) 622-2868.
Midwest: Robert Acker & Assoc., 480 Central Ave., Northfield, IL 60093; (312) 441-6050.
East Coast: Hart Associates, PO Box 339, 42 Lake Blvd., Matawan, NJ 07747; (201) 583-8500.
New England: Arpin Associates, PO Box 227, Weston, MA 02193; (617) 899-5613.
Southeast/Telemarketing: Kay Young & Assoc., 2081 Business Center Dr., Suite 180, Irvine, CA 92714; (714) 551-4924.
Europe: Heinz J. Gorgens, Parkstrasse 8a, D-4054 Nettetal 1 - Hinsbeck (F.R.G.); (0 21 53) 8 99 88.

Advertising Director: Dawn Peck.

For production information, conference, and classified advertising, contact Heidi Rex or Marian Tibayan.

For reader service inquiries, contact Marian Tibayan.

Computer, 10662 Los Vaqueros Circle, Los Alamitos, CA 90720; (714) 821-8380.

PRODUCT INDEX

	RS #	Page #
BOARDS		
Coprocessor	31	134
CIRCUITS		
Accelerator card	30	134
Communications controller	36	136
Floating point processor	9	Cover IV
IC Announcements	120-131	138
COMPONENTS		
Microsystem Announcements	135-146	140
CONFERENCES & COURSES		
Aerospace Computer Security Applications		90
CASE '88		Cover III
Computer Architecture		104
Computer Networking		118
DAC		127
Data Engineering		139
Databases		154
Frontiers '88		121
ICC '88		7
ICCAD '88		89
Infocom '89		10
Medical Systems		141
Security & Privacy		50-51
Sigmatrics		76
DATA COMMUNICATIONS		
LAN	42	137
I/O RELATED EQUIPMENT		
Mouse	20	128
Tablet	21, 22	129, 130
PUBLISHERS		
Computer Society membership		133
Computer Society ombudsman		49
Neural computing booklet		117
Neural network books	2	21
RECRUITMENT		
Classified advertising		149-153
Fellows program	8	159
Technical professional		4-5, 40, 48, 75, 103
SOFTWARE		
Analyst workstation	4	24
C interpreter	24	132
Design tool	5	64
Production tool	43, 44	137
Simulation package		1
SYSTEMS		
Computer	34, 35	135
	37-41	136, 137
Development	33	135
Modeling tool	1, 3	Cover II, 23
Networking	23	131
Neurocomputer	32	134

COMPUTER

March 1988 issue

(Card void after September 1988)

Name _____

Title _____

Company _____

Address _____

City _____ State _____ ZIP _____

Country _____ Phone (____) _____

Please send

(Circle those you want):

- 201 Publications catalog
- 202 Membership information
- 203 Student membership information
- 204 Application for senior membership

Reader interest

(Circle what you liked,
add comments on the back)

- A Guest Editor's Introduction
- B Phonetic Typewriter
- C Adaptive Filtering
- D VLSI Implementation
- E Computing Motion
- F Visual Pattern Recognition
- G ART
- H Structured Neural Nets
- I Self-Organization
- J President's Message
- K Standards
- L Computer Society News
- M Update
- N New Product Reviews
- P New Products
- Q IC/Microsystem Announcements
- R Conferences
- S Calendar/Call for Papers
- T Book Reviews
- U New Literature

Product inquiries

(circle the numbers for products and advertisers
you want more information on)

1	21	41	61	81	101	121	141	161	181
2	22	42	62	82	102	122	142	162	182
3	23	43	63	83	103	123	143	163	183
4	24	44	64	84	104	124	144	164	184
5	25	45	65	85	105	125	145	165	185
6	26	46	66	86	106	126	146	166	186
7	27	47	67	87	107	127	147	167	187
8	28	48	68	88	108	128	148	168	188
9	29	49	69	89	109	129	149	169	189
10	30	50	70	90	110	130	150	170	190
11	31	51	71	91	111	131	151	171	191
12	32	52	72	92	112	132	152	172	192
13	33	53	73	93	113	133	153	173	193
14	34	54	74	94	114	134	154	174	194
15	35	55	75	95	115	135	155	175	195
16	36	56	76	96	116	136	156	176	196
17	37	57	77	97	117	137	157	177	197
18	38	58	78	98	118	138	158	178	198
19	39	59	79	99	119	139	159	179	199
20	40	60	80	100	120	140	160	180	200

2

COMPUTER

March 1988 issue

(Card void after September 1988)

Name _____

Title _____

Company _____

Address _____

City _____ State _____ ZIP _____

Country _____ Phone (____) _____

Please send

(Circle those you want):

- 201 Publications catalog
- 202 Membership information
- 203 Student membership information
- 204 Application for senior membership

Reader interest

(Circle what you liked,
add comments on the back)

- A Guest Editor's Introduction
- B Phonetic Typewriter
- C Adaptive Filtering
- D VLSI Implementation
- E Computing Motion
- F Visual Pattern Recognition
- G ART
- H Structured Neural Nets
- I Self-Organization
- J President's Message
- K Standards
- L Computer Society News
- M Update
- N New Product Reviews
- P New Products
- Q IC/Microsystem Announcements
- R Conferences
- S Calendar/Call for Papers
- T Book Reviews
- U New Literature

Product inquiries

(circle the numbers for products and advertisers
you want more information on)

1	21	41	61	81	101	121	141	161	181
2	22	42	62	82	102	122	142	162	182
3	23	43	63	83	103	123	143	163	183
4	24	44	64	84	104	124	144	164	184
5	25	45	65	85	105	125	145	165	185
6	26	46	66	86	106	126	146	166	186
7	27	47	67	87	107	127	147	167	187
8	28	48	68	88	108	128	148	168	188
9	29	49	69	89	109	129	149	169	189
10	30	50	70	90	110	130	150	170	190
11	31	51	71	91	111	131	151	171	191
12	32	52	72	92	112	132	152	172	192
13	33	53	73	93	113	133	153	173	193
14	34	54	74	94	114	134	154	174	194
15	35	55	75	95	115	135	155	175	195
16	36	56	76	96	116	136	156	176	196
17	37	57	77	97	117	137	157	177	197
18	38	58	78	98	118	138	158	178	198
19	39	59	79	99	119	139	159	179	199
20	40	60	80	100	120	140	160	180	200

3

COMPUTER

March 1988 issue

(Card void after September 1988)

Name _____

Title _____

Company _____

Address _____

City _____ State _____ ZIP _____

Country _____ Phone (____) _____

Please send

(Circle those you want):

- 201 Publications catalog
- 202 Membership information
- 203 Student membership information
- 204 Application for senior membership

Reader interest

(Circle what you liked,
add comments on the back)

- A Guest Editor's Introduction
- B Phonetic Typewriter
- C Adaptive Filtering
- D VLSI Implementation
- E Computing Motion
- F Visual Pattern Recognition
- G ART
- H Structured Neural Nets
- I Self-Organization
- J President's Message
- K Standards
- L Computer Society News
- M Update
- N New Product Reviews
- P New Products
- Q IC/Microsystem Announcements
- R Conferences
- S Calendar/Call for Papers
- T Book Reviews
- U New Literature

Product inquiries

(circle the numbers for products and advertisers
you want more information on)

1	21	41	61	81	101	121	141	161	181
2	22	42	62	82	102	122	142	162	182
3	23	43	63	83	103	123	143	163	183
4	24	44	64	84	104	124	144	164	184
5	25	45	65	85	105	125	145	165	185
6	26	46	66	86	106	126	146	166	186
7	27	47	67	87	107	127	147	167	187
8	28	48	68	88	108	128	148	168	188
9	29	49	69	89	109	129	149	169	189
10	30	50	70	90	110	130	150	170	190
11	31	51	71	91	111	131	151	171	191
12	32	52	72	92	112	132	152	172	192
13	33	53	73	93	113	133	153	173	193
14	34	54	74	94	114	134	154	174	194
15	35	55	75	95	115	135	155	175	195
16	36	56	76	96	116	136	156	176	196
17	37	57	77	97	117	137	157	177	197
18	38	58	78	98	118	138	158	178	198
19	39	59	79	99	119	139	159	179	199
20	40	60	80	100	120	140	160	180	200

Editorial comments

I liked: _____

I disliked: _____

I would like to see: _____

For reader service inquiries, use other side

PO box address for
reader service cards only

PLACE
STAMP
HERE

**Computer
Reader Service Inquiries
PO Box 16508
North Hollywood, CA 91615-6508
USA**

Editorial comments

I liked: _____

I disliked: _____

I would like to see: _____

For reader service inquiries, use other side

PO box address for
reader service cards only

PLACE
STAMP
HERE

**Computer
Reader Service Inquiries
PO Box 16508
North Hollywood, CA 91615-6508
USA**

Editorial comments

I liked: _____

I disliked: _____

I would like to see: _____

For reader service inquiries, use other side

PO box address for
reader service cards only

PLACE
STAMP
HERE

**Computer
Reader Service Inquiries
PO Box 16508
North Hollywood, CA 91615-6508
USA**



CASE '88

Second International Workshop on Computer-Aided Software Engineering

Hyatt Regency Cambridge
Cambridge, Massachusetts
July 12-15, 1988

Cooperation With:

 **THE COMPUTER SOCIETY
OF THE IEEE**

Call For Papers

The field of Computer-Aided Software Engineering (CASE) has recently emerged as a commercially-viable widespread application of software engineering techniques and computer technology to information systems development.

CASE '88 provides a forum for solid, detailed exchange of ideas among key practitioners, researchers, developers, and leading-edge users in the field, resulting in meaningful goals for the advancement of CASE technology in the next five years.

Working groups will review the current state-of-the-art, discuss present research directions, and consider future requirements in the following topic areas:

- Management Issues
 - Management of CASE
 - Technology Transfer
 - Support for Top-level Management & Strategic Planning
- Methods for Information Systems Development
 - Integration of Methods
 - Support between Methods & Tools
 - Real-time Development Methods
 - Object-Oriented Design
- Enabling Technologies
 - A.I. / Expert Systems
 - Programming Language Evolution
 - Database / Knowledge Base Architectures
 - Ada™ Technology
 - User Working Environment / CHI
- Reverse Engineering
- Standards for CASE

General Chair

Elliot Chikofsky
Index Technology Corporation

Program Committee

Ronald J. Norman
San Diego State University

Hasan Sayani
Advanced Systems Technology Corporation

Jerrold Grochow
American Management Systems

John O. Jenkins
Imperial College

Burt Rubenstein
Index Technology Corporation

Tony Wasserman
Interactive Development Environments

Thomas P. Cullinane
Northeastern University

Karl Lieberherr
Northeastern University

Bob Carasik
Pacific Bell

Peter Mager
PSM Associates

Lonnie Bentley
Purdue University

Priscilla Fowler
Software Engineering Institute

William Cureton
Sun Microsystems

Kate Ehrlich
Symbolics

Alan R. Hevner
University of Maryland

W. Richards Adrion
University of Massachusetts

Lt. Michael F. Merriman
U.S. Air Force ESD

Thomas Browdy
Washington University

Attendance is limited. Prospective attendees should submit 5 copies of a position paper (1 to 4 pages), discussion forum proposal, or extended abstract (up to 10 pages) on future directions of CASE technology related to one or more of the topic areas, **by April 11, 1988.**

Machine-readable submissions are preferred.

Program Coordinator

Pamela Meyer
CASE '88
c/o Index Technology Corporation
One Main Street
Cambridge, MA 02142
(617) 494-8200 ext. 1988

CSNET: Chikofsky@VAXE.COE.
Northeastern.EDU

European Coordinator

John O. Jenkins
Imperial College
School of Management
London SW7 2PG, United Kingdom
01-589-5111 ext. 7112

Exhibits Chair

Victor Barlow
Purdue University
Dept. of Computer Technology
Knoy Hall, Room 242
West Lafayette, IN 47907
(317) 494-4546

Sponsored by:
Index Technology Corporation

In Cooperation With:
Imperial College
Purdue University
Northeastern University
Washington University CSDP
Greater Boston Chapter ACM



FLOATING POINT PROCESSOR

22 MFlops (Peak) PC/AT Card
12 MBytes RAM on board
Pipelined Harvard Architecture
32 and 64 bit integer
32 and 64 bit IEEE floating point
C Compiler
Macro Assembler

***For Real World
Artificial Neural
Systems Applications***



Delta FPP™ PC/AT Card \$14,950
3.1M PEs and Wts.
11M Connections/sec w/o learning
2.7M Connections/sec w/BP learning

Software:

ANSkit™ 13 neural models	\$995
ANSpec™ ANS high order language	\$2,995
C Compiler	\$395
Macro Assembler	\$150



**Science Applications
International Corporation**
An Employee-Owned Company

10260 Campus Point Dr., MS 34, San Diego, CA 92121, (619) 546-6290

FAX (619) 546-6777